University of California
Santa Barbara

Modeling and Calibrating the Distributed Camera

A dissertation submitted in partial satisfaction
of the requirements for the degree

Doctor of Philosophy
in
Computer Science

by

Christopher M. Sweeney

Committee in charge:

Professor Tobias Höllerer, Co-Chair
Professor Matthew Turk, Co-Chair,
Professor Pradeep Sen
Professor Noah Snavely

March 2016

The Dissertation of Christopher M. Sweeney is approved.

Professor Pradeep Sen

Professor Noah Snavely

Professor Tobias Höllerer, Committee Co-Chair

Professor Matthew Turk, Committee Co-Chair

January 2016

Modeling and Calibrating the Distributed Camera

Acknowledgements

I wish to thank my parents and my siblings for their constant love and support. My accomplishments simply would not have been possible without them.

I thank my advisors, Matthew Turk and Tobias Höllerer, for their incredible guidance, patience, and mentoring throughout my graduate studies. I have learned a great deal about computer vision, geometry, and how to conduct research from them. They have consistently put me in a position to achieve my goals, and I sincerely appreciate the all of the conversations and time spent together.

I would also like to thank Marc Pollefeys for inviting me to visit the Computer Vision and Geometry group at ETH Zürich, and to my friend and colleague Torsten Sattler for arranging my visit and for being a constant resource for me. I thank my lab mates at the Four Eyes Lab and at ETH for an uncountable number of brainstorming sessions, conversations, and debugging sessions over the past several years, especially Jonathan, Steffen, Victor, Ben, Bernhard, Laurent, and Matthias. I also want to thank John Flynn, James Philbin, and Sameer Agarwal for their excellent mentorship.

Finally, I would like to thank all of my friends in Santa Barbara, Matt, Mike R., Ed, Mike Z., Michelle, Mary Claire, Whitney, Steve, and most importantly Julia, for all of their support and understanding throughout my studies. They are a major reason why my time in Santa Barbara has been so special and a time that I will forever cherish.

Chris Sweeney

(571) 334-6185
cmsweeney@cs.ucsb.edu
1546 Shoreline Drive
Santa Barbara, CA 93109

EDUCATION

Doctor of Philosophy, Computer Science, Conferred March 2016
University of California, Santa Barbara, Santa Barbara, CA
Dissertation: Modeling and Calibrating the Distributed Camera
Advisors: Tobias Höllerer and Matthew Turk

Visiting PhD Student, Computer Vision and Geometry Lab, October 2014 – April 2015
Eidgenössische Technische Hochschule (ETH) Zürich, Zürich, Switzerland
Hosted by: Professor Marc Pollefeys, Dr. Torsten Sattler

Bachelor of Science (with High Distinction), Computer Science
Bachelor of Arts, **Mathematics**
May 2011, University of Virginia, Charlottesville, VA

RESEARCH AND PROFESSIONAL EXPERIENCE

Graduate Research Assistant, 4 Eyes Research Lab, Department of Computer Science, University of California, Santa Barbara
Santa Barbara, CA, Sept 2011 – 2016

Research Intern, Lightfield Group, Google, Inc, Seattle, WA
Summer 2014, Under the supervision of Sameer Agarwal

Research Intern, Visual Search Team, Google, Inc, Los Angeles, CA
Summer 2012, Summer 2013

Undergraduate Research Assistant, Department of Computer Science, University of Virginia, Charlottesville, VA
August 2009 - May 2011

HONORS/AWARDS

ISMAR 2015 Best Short Paper
Winner of ACM Open Source Software Competition (2015)
National Science Foundation (NSF) Graduation Research Fellowship, 2013 - 2016
ISMAR 2012 Best Paper Award
Graduate Opportunity Fellowship (University of California, Santa Barbara), 2011-2012
Louis T. Radar Award for Research (University of Virginia), 2011
Voted Best Internship Project (Google, Inc), 2010, 2012
Google Outstanding Undergraduate Engineering Scholarship, 2010-2011
Member of Raven Society (University of Virginia)
Computing Research Association Outstanding Undergraduate Research Award, 2011
NSF PAGES Fellowship, 2007-2011

PUBLICATIONS

L. Kneip, **C. Sweeney,** R. Hartley, "The Generalized Relative Pose-and-Scale Problem: View-Graph Fusion via 2D-2D registration". Winter Conference on Applications of Computer Vision (WACV). 2016.

C. Sweeney, T. Sattler, T. Höllerer, M. Turk, M. Pollefeys, "Optimizing the Viewing Graph for Structure-from-Motion", Proceedings of the International Conference on Computer Vision (ICCV), 2015.

C. Sweeney, T. Höllerer, M. Turk, "Theia: A Fast and Scalable Structure-from-Motion Library", ACM Open Source Software Competition. 2015. **[Competition Winner]**

C. Sweeney, J. Flynn, B. Nuernberger, M. Turk, T. Höllerer, "Efficient Computation of Absolute Pose for Gravity-Aware Augmented Reality", Proceedings of the IEEE Symposium on Mixed and Augmented Reality (ISMAR), 2015. **[Best Short Paper Award]**

C. Sweeney, L. Kneip, T. Höllerer, M. Turk, "Computing Similarity Transformations from Only Image Correspondences", Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

C. Sweeney, J. Flynn, M. Turk, "Solving for Relative Pose with Partially Known Rotation is a Quadratic Eigenvalue Problem", Proceedings of the International Conference on 3D Vision (3DV), 2014.

C. Sweeney, V. Fragoso, T. Höllerer, M. Turk, "gDLS: A Scalable Solution to the Generalized Pose and Scale Problem", Proceedings of the European Conference on Computer Vision (ECCV), 2014.

T. Sattler, **C. Sweeney,** M. Pollefeys, "On Sampling Focal Length Values to Solve the Absolute Pose Problem", Proceedings of the European Conference on Computer Vision (ECCV), 2014.

S. Gauglitz, **C. Sweeney**, J. Ventura, M. Turk, T. Höllerer, "Model Estimation and Selection towards Unconstrained Real-time Tracking and Mapping", IEEE Transactions on Visualization and Computer Graphics (TVCG), vol. 20, no. 6, pp. 825-838, June 2014

C. Sweeney, M. Turk, T. Höllerer, "Improved Outdoor Augmented Reality through 'Globalization'", Doctoral Consortium at the International Symposium on Mixed and Augmented Reality (ISMAR), 2013

S. Gauglitz, **C. Sweeney**, J. Ventura, M. Turk, T. Höllerer, "Live Tracking and Mapping from Both General and Rotation-Only Camera Motion", International Symposium on Mixed and Augmented Reality (ISMAR), 2012 **[Best Paper Award]**

C. Sweeney, L. Liu, S. Arietta, J. Lawrence, "HIPI: A Hadoop Image Processing Interface for Image-based MapReduce Tasks", Undergraduate Senior Thesis, School of Engineering and Applied Sciences, University of Virginia, 2011

TEACHING EXPERIENCE

Graduate Reader, Mixed and Augmented Reality (CS 290I), Department of Computer Science, University of California, Santa Barbara, Santa Barbara, CA
Winter 2012

- Responsible for creating and grading all homework assignments
- Gave several lectures on mobile augmented reality

Teaching Assistant, Program and Data Representations (CS 216), Department of Computer Science, University of Virginia, Charlottesville, VA
Spring 2010

- Ran 2 hour lab lecture once per week
- Held weekly office hours, met with students one-on-one
- Responsible for grading homework assignments and exams

OPEN SOURCE CONTRIBUTIONS

Theia: an open source library for structure-from-motion
URL: http://www.theia-sfm.org

- 2015 ACM MM Open Source Software Competition Winner
- Primary developer, maintainer
- Provides a flexible, efficient end to end system for researchers and users in Computer Vision

HIPI: Hadoop Image Processing Library
URL: http://hipi.cs.virginia.edu/

- Primary developer, maintainer
- Created custom distributed file storage for large-scale images
- Designed a research tool customized for image processing and computer vision applications

Libjpeg Turbo

- Created bit-operative Huffman encoding with parity aware recovery

PROFESSIONAL ACTIVITIES

Invited Lectures

- Carnegie Mellon VASC Seminar Series, September 2015
- University of Washington, July 2015
- Microsoft Research, July 2015
- TU Graz, March 2015
- University of Virginia, September 2014

Invited Reviewer

- IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
- IEEE International Conference on Computer Vision (ICCV)
- European Conference on Computer Vision (ECCV)
- ACM Special Interest Group on Computer Graphics (SIGGRAPH)
- IEEE International Symposium on Mixed and Augmented Reality (ISMAR)
- IEEE Transactions on Visualization and Computer Graphics (TVCG)
- IEEE Virtual Reality (VR)
- Journal of Image and Vision Computing

ACADEMIC ACTIVITIES

University Services

- Madison House Alumni Council, Vice Chair (2012 – present)
- University of Virginia Young Alumni Council (2012 – 2014)
- Graduate Student Association Department Representative (2012 – 2014)

Abstract

Modeling and Calibrating the Distributed Camera

by

Christopher M. Sweeney

Structure-from-Motion (SfM) is a powerful tool for computing 3D reconstructions from images of a scene and has wide applications in computer vision, scene recognition, and augmented and virtual reality. Standard SfM pipelines make strict assumptions about the capturing devices in order to simplify the process for estimating camera geometry and 3D structure. Specifically, most methods require monocular cameras with known focal length calibration. When considering large-scale SfM from internet photo collections, EXIF calibrations cannot be used reliably. Further, the requirement of single camera systems limits the scalability of SfM.

This thesis proposes to remove these constraints by instead considering the collection of cameras as a *distributed camera* that encapsulates the image and geometric information of all cameras simultaneously. First, I provide full generalizations to the relative camera pose and absolute camera pose problems. These generalizations are more expressive and extend the traditional single-camera problems to distributed cameras, forming the basis for a novel hierarchical SfM pipeline that exhibits state-of-the-art performance on large-scale datasets. Second, I describe two efficient methods for estimating camera focal lengths for the distributed camera when calibration is not available. Finally, I show how removing these constraints enables a simpler, more scalable SfM pipeline that is capable of handling uncalibrated cameras at scale.

Contents

Abstract **viii**

1 Introduction **1**
1.1 Overview of Contributions 8
1.2 Thesis Organization 9

2 Foundations **11**
2.1 The Pinhole Camera Model 11
2.2 Structure-from-Motion 15

Part I Modeling the Distributed Camera **25**

3 A Full Generalization of the Relative Pose Problem **26**
3.1 A Quadratic Eigenvalue Formulation 28
3.2 Generalization to 7 d.o.f. 32
3.3 Experimental Evaluation 35
3.4 Discussion 42

4 A Full Generalization of the Absolute Pose Problem **45**
4.1 Generalization to 7 d.o.f. 47
4.2 An L_2 Optimal Solution 49
4.3 Experimental Evaluation 56
4.4 A Hierachical SfM Pipeline 65
4.5 Discussion 68

Part II Calibrating the Distributed Camera **70**

5 Computing the Focal Length of a Single Camera **71**
5.1 Problem Formulation 73
5.2 Probabilistic Focal Length Sampling 76

5.3 Image-based Localization Evaluation 84
5.4 Discussion 88

6 Computing Camera Focal Lengths at Scale 91
6.1 Focal Lengths from a Fundamental Matrix 92
6.2 Focal Lengths from the Viewing Graph 94
6.3 The Viewing Graph 95
6.4 Creating a Consistent Viewing Graph 96
6.5 Estimating Structure and Motion 100
6.6 Experimental Evaluation 105
6.7 Discussion 109

Part III Open Source Contributions and Conclusions 112

7 Theia: A Fast and Scalable Structure-from-Motion Library 113
7.1 Overview of Features 114
7.2 SfM Pipeline 120
7.3 Impact 123

8 Conclusion 124
8.1 Future Work 126

A Computing Matrices U, S, and V for Depth, Scale, and Translation 129

B Triplet Projection Error 132

Bibliography 134

Chapter 1

Introduction

The world around us is rich with visual content. There are countless objects, shapes, light sources, materials, and textures that our eyes continuously observe as we move about the world. Recreating the rich, continuous stream of visual information that our eyes provide is a difficult task. Cameras have the ability to capture a particular place at a particular moment in time and in this way provide a snapshot into the world that we observe. A photograph can invoke a wide variety of emotions and memories, and viewers can be transported to the instant that the photograph was originally taken. Since the advent of film in the late 19th century, cameras have been used as a cheap and simple way to capture memorable moments and places. We capture moments such as birthday parties, sporting events, wildlife, and natural landscapes and are able to revisit them at any time through photographs. Before the digital revolution of the past 20 years, the number of photos that one could capture was limited by the capacity of the film roll. With the advent of the digital camera and continually decreasing storage costs we are now able to capture large quantities of images at a low cost. But with our ever-growing digital catalog of images comes the question: are we actually gaining a better experience with all these new images?

When considered on its own an image provides only a tiny glimpse of the scene in front of the photographer. This experience can be quite powerful but it is inherently

Figure 1.1: Structure-from-Motion is able to recover a 3D models of world landmarks such as the Taj Mahal and Mount Rushmore. Viewers are able to freely explore these 3D models to oberve the scene from any angle and position to gain a better understanding of the scale and structure. This enables viewers to observe the scene from novel viewpoints that may not be easily accessible in the real world.

limited by the pixels that comprise the image, and we lack the ability to explore the scene or moment captured beyond those pixels. In contrast, the real world is rich with visual information because we experience depth, perspective, and freedom of motion to explore areas of interest. To fully appreciate a world landmark such as the Taj Mahal (*cf.* Figure 1.1), we would like to walk around its gardens, stroll past the reflection pools, and view the detail of the marble construction up close. Can the beauty of landmarks such as the Taj Mahal really be captured in a single image? If we truly seek to explore the snapshot of space and time captured by a photograph, more information is needed. This is especially true for large, complex spaces such as Mount Rushmore where the scale or structure may be exceedingly difficult to understand from images alone (see Figure 1.1).

Given the proliferation of images on the internet through websites such as Flickr, Google Photos, and Facebook, we now have access to an unprecedented number of rich and diverse images of many cities and landmarks around the world. If we seek to explore

Figure 1.2: Flickr allows landmarks such as Notre Dame may be explored as scrollable image grids with hundred of thousands of user-submitted photos. While the range of images is diverse, navigating large quantities of images is both unfulfilling and unfeasible for users interested in experiencing what it is like to be present at Notre Dame.

the Notre Dame Cathedral in Paris, for instance, we can simply search "Notre Dame" on Flickr and are provided with hundreds of thousands of free, publicly available images of Notre Dame (*cf*. Figure 1.2). These images include a wide range of positions and viewpoints captured in varying seasons, times of day, and weather patterns that provide a detailed visual catalogue of the scene. This variety and diversity provides contextual clues for understanding a scene beyond what a single image would allow.

While additional images can help our understanding of a scene, they do not necessarily improve the feeling of actually experiencing a place. Worse, navigating a large collection of images is a difficult task since the number of images that can be viewed at one time is limited by the size of the display (*cf*. Figure 1.2). By considering the entire collection of images, however, we are able to obtain a better understand the full size, structure, and detail of scenes such as the Notre Dame Cathedral. By using omnidirectional cameras, Google Street View allows for virtual "walk-throughs" along streets to

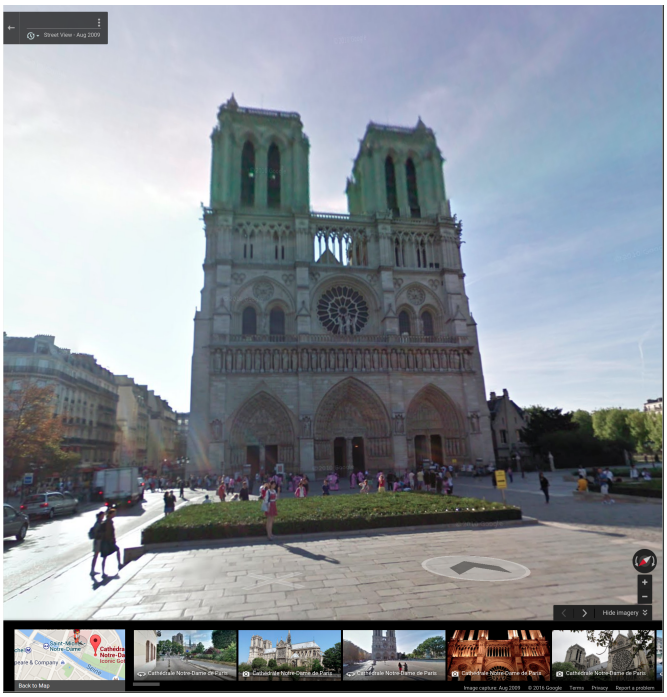

Figure 1.3: Google Street View provides virtual "walk-throughs" that allow the user to click on arrows to move throughout the scene.

simulate the experience of being there (*cf*. Figure 1.3); however, this requires expensive, specialized camera rigs, a laborous capture process, and time-consuming post-processing. Similarly, multi-camera rigs such as Google Jump and Nokia Ozo capture high resolution 360° video that can be used for playback in virtual reality. These camera rigs provide excellent immersion for viewers, but cost tens of thousands of dollars, making them infeasible for every-day consumers.

What if instead of viewing the collection of images as many individual snapshots we consider each image to be a new observation from a *distributed camera* that observes the scene from many perspectives simultaneously? This distributed camera may contain many observations taken at many different locations that provide unique snapshots of the scene. If we knew the location of these observations then we could use the observations to understand and explore the scene better. Fortunately, computers excel at a task that humans cannot easily perform: recovering precise 3D geometry from images alone.

Research in the field of multiple view geometry has allowed for automatic recovery of camera and 3D scene geometry through a process called *Structure-from-Motion* (SfM) [33, 81]. The output of SfM is a *3D point cloud* with correspondence information describing which 3D points are observed in which images along with the position and orientation of images used to create the 3D reconstruction. In this way, the output of SfM can be thought of as the distributed camera along with 3D points, and the 3D reconstruction process can be thought of as recovering the configuration of the distributed camera. This 3D representation is not only more compact than image information, it is also rich with visual information about the entire scene as the 3D point clouds contain local and global appearance information in addition to 3D structure. By recovering 3D geometry, large and complex scenes may be viewed in a more natural manner with 3D visualization tools such as Meshlab that allow for unconstrained visualization and exploration. This thesis focuses on 3D modeling with SfM so that we may better understand scenes through geometry.

By utilizing the rich source of visual data from websites such as Flickr, Facebook, and Google Photos, we are able to apply SfM to photo collections of popular landmarks around the world to create visually rich 3D models that can be readily explored. Performing SfM on internet photo collections was first proposed in the seminal Photo Tourism work [73], and has since been incorporated into products such as Microsoft's Photosynth and Google's Photo Tours. To create 3D models with SfM, correspondences between images are established by matching salient regions called *features* within the images. Then, two images with a large number of feature correspondences are used to create an initial 3D reconstruction by computing their. The initial relative position and orientation of the two images is determined by the location of their feature correspondences (*i.e.*, the *relative camera pose problem*). Matching features in each image pair are used to triangulate 3D points, producing an initial 3D point cloud. Images that observe these 3D points

are incrementally localized to this initial model (*i.e.*, the *absolute camera pose problem*) and more 3D points are triangulated until all possible images have been added to the 3D model. This process is called *incremental SfM* and has proven to be robust and accurate for a wide variety of scenes; however, the sequential nature of incremental SfM does not scale well, limiting the size of the 3D models that can be created.

Whereas incremental SfM methods consider one image at a time, *global SfM* methods consider all images simultaneously when creating a 3D model. Considering all images simultaneously allows for each image to be weighed equally when creating the 3D model, unlike incremental SfM where the order in which images are added to the growing reconstruction affects the final quality of the reconstruction (*i.e.*, images added earlier are given more weight). Efficient linear and nonlinear solvers exist for global SfM [20] [22] [29] [38] and, as a result, these methods are more readily parallelized with existing optimization libraries and can be orders of magnitude more efficient than incremental SfM methods on large-scale datasets. While global SfM methods are promising for large-scale SfM, their use is limited because they utilize Euclidean motion averaging algorithms and thus require accurate camera calibration information of each camera's focal length. For internet photo collections, camera calibration is often unavailable or inaccurate and as a result global SfM methods can be less accurate and robust than incremental methods on these datasets.

Ideally, we would like to have the robustness of incremental SfM methods and the efficiency and scalability of global SfM methods when creating 3D models. Incremental and global SfM methods, however, are limited in scope because of two strong assumptions. First, both global and incremental SfM methods assume that images were taken with individual cameras. This limits the scalability of both methods, especially incremental SfM where current camera localization methods only allow for one image to be added to the reconstruction at a time. If we utilize the distributed camera as a building block for

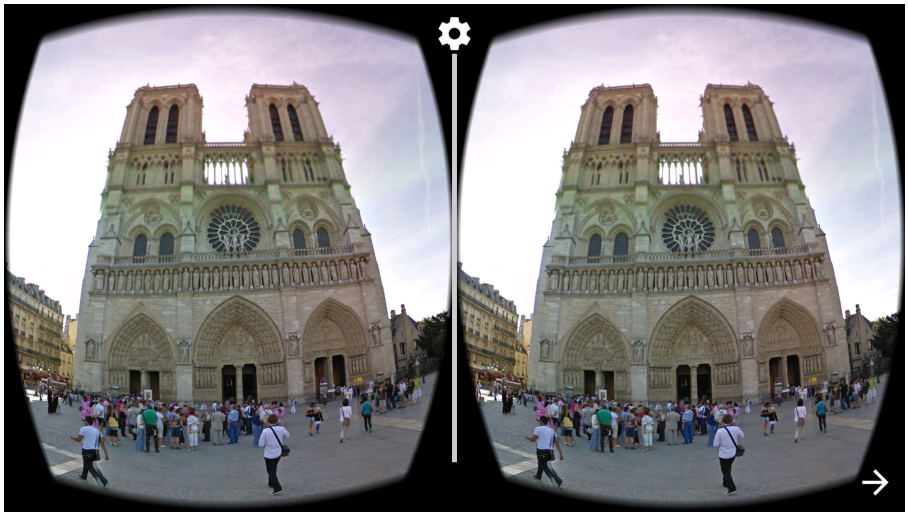

Figure 1.4: Virtual reality displays such as Google Cardboard and Oculus Rift can provide users with stereoscopic views of Notre Dame, allowing users to freely look around the scene while experiencing important visual cues such as depth and occlusion.

SfM instead of merely an output, we can dramatically improve the scalability of SfM by localizing all images within the distributed camera at once. Second, global SfM requires very accurate knowledge of camera intrinsic parameters such as focal length and principal point. Without accurate camera intrinsics the accuracy of global SfM methods suffers dramatically, limiting the usability of such methods on internet photo collections where camera intrinsic information may be inaccurate or unavailable.

In this thesis, I focus on how to remove these assumptions so that SfM may be more readily used to obtain high-quality large-scale 3D models. In particular, I focus on scalable and efficient methods that must also be accurate and robust. By crowd-sourcing the creation of accurate 3D models through internet photo collections, viewers may experience large and complex scenes in a compelling way that images alone would not allow. Such large scenes would be especially interesting to visualize and explore in virtual reality viewers such as Oculus Rift[1] or Google Cardboard[2]. Removing assumptions

[1] `https://www.oculus.com`

[2] `https://www.google.com/get/cardboard/`

of single-camera models and intrinsic calibration from SfM allows for a complete and general expression of the distributed camera, leading to simple and scalable methods for recovering camera motion and 3D structure from calibrated or uncalibrated image sets.

1.1 Overview of Contributions

I address two key limitations of current large-scale SfM algorithms in this thesis: the assumption that each image was taken with a single perspective camera, and the assumption that accurate camera calibration information is available.

Modeling the distributed camera. I remove the assumption of single perspective cameras by presenting complete mathematical generalizations to two fundamental problems in SfM: the relative camera pose problem and the absolute camera pose problem. These generalizations extend the standard single-camera and multi-camera relative and absolute pose problems to 7 degrees-of-freedom (d.o.f.) similarity transformations, and I show that the single-camera and multi-camera relative and absolute pose problems are special cases of this general formulation. This formulation is capable of considering many cameras at once in addition to scale changes and thus can be used directly on distributed cameras. Further, these novel formulations provide new insights into model merging, including estimating 7 degrees-of-freedom similarity transformations that minimize reprojection error.

Calibrating the distributed camera. Accurate camera calibration is of great importance to SfM. For global SfM, obtaining accurate calibration of camera intrinsic parameters is essential to obtain high quality 3D models. Unfortunately, camera calibration is often unavailable or inaccurate when utilizing internet photo collections as input for SfM, and few methods exist to accurately calibrate images. To this end, I propose two efficient methods for estimating camera focal lengths. The first method estimates

camera focal lengths along with the camera pose using only 3D geometry from an existing SfM reconstruction. The second method utilizes a network of cameras to calibrate many cameras simultaneously from the 2-view matches between images. This calibration method is then combined with a viewing graph optimization that is able to dramatically improve the quality of the relative geometries used for SfM, leading to a simple global SfM pipeline that is able to operate on uncalibrated image sets and improves the efficiency of global SfM up to one order of magnitude.

Open source software contributions. Few open-source software libraries for SfM exist, and most of the available libraries only implement incremental SfM. When I began my graduate studies, no libraries implementing global SfM existed. To address this, I created the Theia Structure-from-Motion library that implements highly customize-able end-to-end SfM pipelines. The library was designed with code that is clean, modular, and extendable with thorough unit-testing. Further, the library was designed to handle large-scale SfM and is optimized for memory efficiency and parallelism. With Theia, researchers may easily tweak and experiment with specific parts of global or incremental SfM pipelines without having to worry about implementing an entire SfM pipeline from scratch.

1.2 Thesis Organization

The remainder of this thesis is divided into three parts. First, Chapter 2 introduces fundamental concepts of multiple view geometry including the pinhole camera model and an overview of incremental and global SfM pipelines. Part I considers how to remove the assumption that every image in SfM is taken with a single-perspective camera in order to model the distributed camera. We view this limitation through the lens of two classic problems: the relative camera pose problem (Chapter 3) and the absolute camera pose

problem (Chapter 4). Part II discusses effective methods for calibrating camera intrinsic parameters for SfM and thus the distributed camera. Chapter 5 discusses an efficient and accurate method for calibrating an image from an existing 3D model. In Chapter 6 I present techniques for estimating focal lengths at scale from a network of cameras. This technique, when combined with a pre-SfM optimization, allows for global SfM to be performed on uncalibrated image sets. Finally, in Part III, I discuss my open-source SfM library, Theia (Chapter 7), and review the contributions of this thesis and provide concluding notes (Chapter 8).

Chapter 2

Foundations

This chapter reviews some of the core concepts and fundamental theories for multiple view geometry and SfM that are relevant to this thesis. In Section 2.1, we review the *pinhole camera model* which describes a mathematical representation for cameras in geometric computer vision. In particular, this model maps points in 3D space to pixels in image space and is a core construct for SfM. Since Part I of this thesis discusses new methods for 3D modeling, Section 2.2 provides an overview of incremental and global SfM procedures used to create 3D models from sets of images.

2.1 The Pinhole Camera Model

The *pinhole camera model* is a widely used mathematical model to describe how cameras observe the 3D space captured in images. Formally, it provides a geometric relationship between a 3D point and its projection onto the image plane of a camera [33]. The mathematics utilize an *ideal* pinhole camera, assuming the camera aperture is an infinitely small point and no lenses are used to focus light. Because it does not account for lens distortion artifacts such as radial or tangential distortion, the pinhole camera model is only an approximation of a real camera and the accuracy of this approximation descreases from the center of the image outwards as lens distortion increases [33]; how-

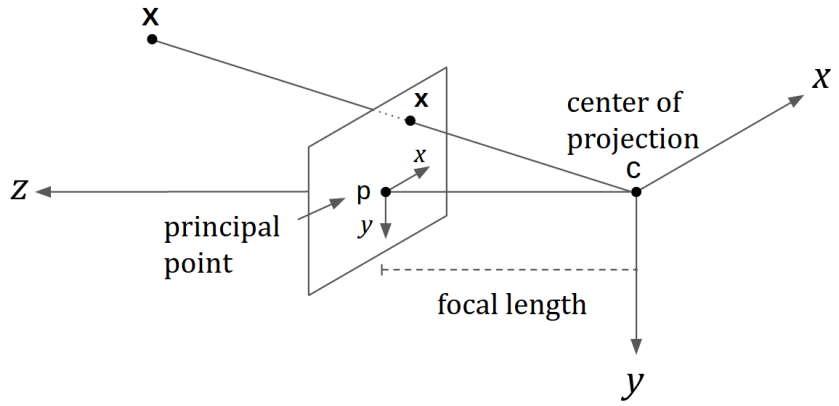

Figure 2.1: The pinhole camera model corresponds a pixel $\mathbf{x}$ to the location where the ray from the center of projection C to the 3D point $\mathbf{X}$ intersects the imaging plane. This dissertation uses a right-handed coordinate system where the camera is looking down the positive Z-axis with the positive X-axis pointing to the right.

ever, for cameras of sufficient quality (such as those commonly used in internet photo collections) this model is a reasonable approximation. For this thesis it is assumed that the effects of lens distortion are negligible or have been previously removed.

2.1.1 Camera Intrinsic Parameters

The pinhole camera model defines a projection function that maps 3D points in a scene to 2D pixel positions in an image. This projection function is modelled as rays that extend from the 3D points in the scene through a *center of projection* [33] (*i.e.*, the pinhole) and intersects with the *image plane* (*cf*. Figure 2.1). The point at which the ray intersects the image plane corresponds to a pixel location where that 3D point is observed in the image. Note that this model allows for continuous pixel values, whereas real image sensors have discrete pixel locations. The distance from the image plane to the center of projection is called the *focal length* of the image, and the point projecting the center of projection onto the image plane is the *principal point*. These values, along with pixel skew and pixel aspect ratio, define the *intrinsic camera calibration matrix*. In

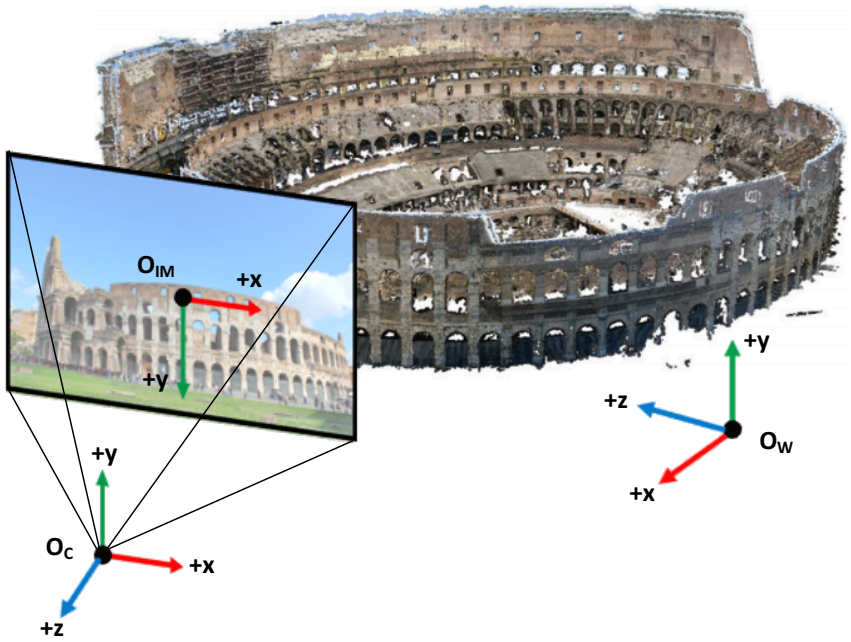

Figure 2.2: Camera intrinsic parameters transform points from the camera coordinate system O_C to the image coordinate system O_{IM}, whereas the extrinsic parameters transform points in the world coordinate system O_W to the camera coordinate system O_C.

its most general form, the intrinsic camera calibration matrix encompasses focal length f, principal point $(p_x,\ p_y)$, aspect ratio a, and skew s such that the projection of a 3D point in the *camera coordinate system* to the 2D pixel location in the *image coordinate system* (*cf*. Figure 2.2) is defined by the matrix

$$K = \begin{bmatrix} f & s & p_x \\ 0 & a \cdot f & p_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.1}$$

Nearly all modern cameras have square pixels with no skew, so we assume that $s = 0$ and $a = 1$ for all cameras. We additionally assume that the principal point is at the center of the image, a reasonable assumption for modern cameras[81]. Thus, in this thesis a

reduced intrinsic camera calibration matrix is used:

$$K = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.2}$$

This matrix may be applied to map a 3D point $\mathbf{X_{cam}} = (X,\ Y,\ Z)^\top$ to a 2D pixel location $\mathbf{x_{im}} = (x,\ y)^\top$ such that the homogeneous pixel location of the image is:

$$\mathbf{x_{homog}} = \begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = K \cdot \mathbf{X_{cam}} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} f \cdot X \\ f \cdot Y \\ Z \end{bmatrix} \tag{2.3}$$

assuming the position of $\mathbf{X_{cam}}$ is given in the camera's coordinate system. A division by the third entry converts homogeneous coordinates to pixel coordinates, resulting in $\mathbf{x_{im}} = (f \cdot X/Z,\ f \cdot Y/Z)^\top$.

2.1.2 Camera Extrisic Parameters

To this point, the 3D point $\mathbf{X}$ has been provided in the same coordinate system as the camera. Yet, it is often the case that the location 3D points are provided in reference to a *world coordinate system* (*cf*. Figure 2.2) such as a landmark. The transformation from a world coordinate system to a camera coordinate system may be defined by a rotation and a translation. This rotation and translation comprise the *camera extrinsic parameters* and this transformation may be written linearly as

$$\mathbf{X_{cam}} = [R \mid t] \cdot \mathbf{X_{world}} \tag{2.4}$$

where R is the 3×3 rotation matrix and t is the translation that transforms a point in the world coordinate system to the camera coordinate system. The rotation R is referred to as the *camera orientation* and the *camera position* may be easily recovered as $\mathbf{C} = -R^\top \cdot t$.

In order to transform a 3D point in world coordinates to a 2D pixel observation in image coordinates, we must apply the extrinsic and intrinsic camera paramters to transform the point:

$$\mathbf{x_{im}} = K \cdot \mathbf{X_{cam}} = K \cdot [R \mid t] \cdot \mathbf{X_{world}}. \tag{2.5}$$

This defines the full projection from a 3D point in world coordinates to pixel locations in image coordinates.

2.2 Structure-from-Motion

The techniques presented in this thesis are centered around recovering 3D geometry of a scene. Structure-from-Motion (SfM) is the process of recovering this 3D geometry from image observations. When SfM is applied to large, complex scenes it can produce beautiful, intricate 3D models that reveal detail, structure, and scale that is not otherwise possible to observe from images alone. In this section, I briefly describe the the fundamental building blocks of SfM and the general procedure for the two most common types of SfM pipelines: incremental SfM, and global SfM.

2.2.1 Epipolar Geometry

In order to recover 3D structure from a set of images, we must first determine the *epipolar geometry* between pairs of cameras. Epipolar geometry describes the geometric

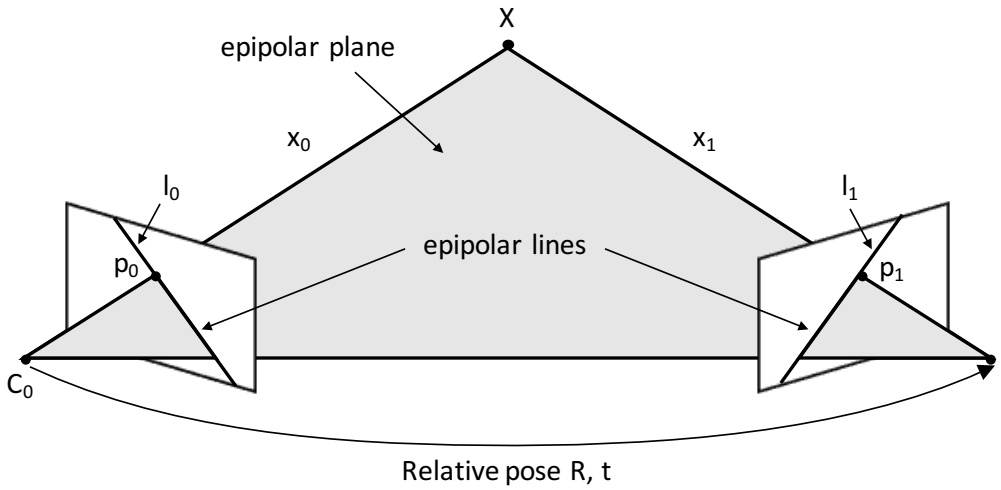

Figure 2.3: Epipolar geometry defines the relationship between two cameras with centers C_0 and C_1 with a known relative pose R, and t. The fundamental or essential matrix maps the image observation p_0 in the left image to the line l_1 in the right image as the intersection of the image plane with the epipolar plane formed by C_0, C_1, and X.

relationship between a pair of images, and can be used to compute the *relative pose* comprised of a relative rotation R and relative translation direction t. Since we are determining the relative pose between the cameras, the first camera is generally considered to be fixed at the origin with an identity rotation so the relative pose is equivalent to the extrinsic parameters of the second camera. Note that due to scale ambiguity, the translation direction may be stretched to any length without altering the epipolar geometry. Figure 2.3 demonstrates the epipolar geometry between two cameras in a scene.

The epipolar geometry may be derived directly from the projection constraints. In C_0's coordinate system, the point X may be represented in terms of the unit-norm ray x_0 (corresponding to the pixel p_0) and the depth of the point α_0 such that $X = \alpha_0 x_0$. Similarly, X may be represented in C_1's coordinate system as:

$$\alpha_1 x_1 = R(\alpha_0 x_0) + t \tag{2.6}$$

By taking the cross product of t with both sides we obtain:

$$t \times \alpha_1 x_1 = [t]_\times R(\alpha_0 x_0), \tag{2.7}$$

where $[t]_\times$ is the skew-symmetric cross product matrix:

$$[t]_\times = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} \tag{2.8}$$

We may eliminate the left-hand side of the equation by taking the dot product with x_1:

$$0 = x_1^\top ([t]_\times R)(\alpha_0 x_0). \tag{2.9}$$

Finally, we may drop the scalar α_0 (as it does not effect the constraint) to obtain the *epipolar point constraint*:

$$0 = x_1^\top ([t]_\times R) x_0 = x_1^\top E x_0. \tag{2.10}$$

The 3×3 rank-2 matrix E is called the *essential matrix*, and encapsulates the relative pose between the two cameras. This matrix maps a point in the first image to the *epipolar line* in the second image $l_1 = Ex_0$ by projecting the ray x_0 from the first image onto the line l_1 in the second image. Additionally, the epipolar line is the intersection of the *epipolar plane* formed by C_0, C_1, and X with the image plane of the second view. The same procedure may be applied to derive the epipolar constraints from the second image to the first image.

If five correspondences between points are known, the essential matrix may be re-

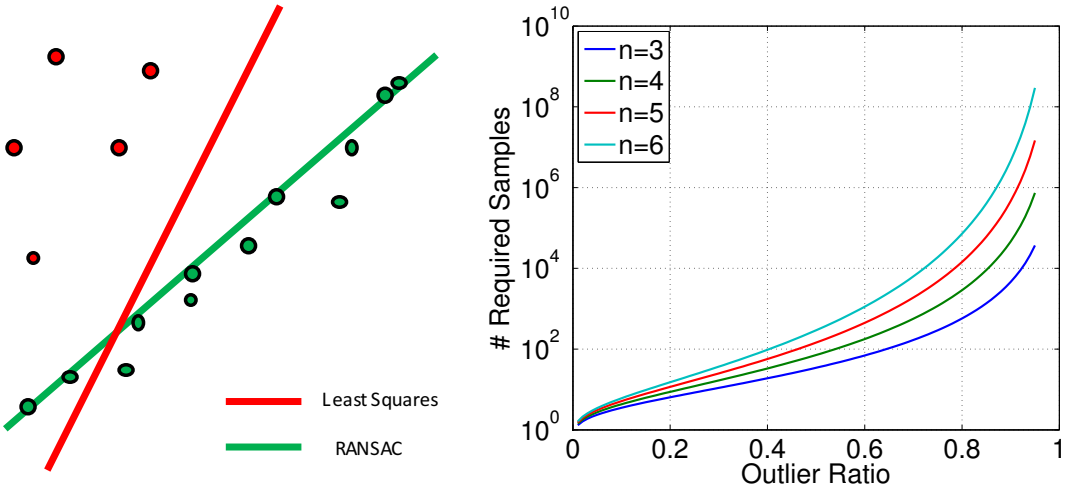

Figure 2.4: **Left:** RANSAC is able to robustly estimate the line of best fit from inliers (green dots) despite the presence of outliers (red dots) in the data. **Right:** The number of iterations required to obtain an all-inlier ratio with 99% confidence increases exponentially as the number of data points per sample n increases.

covered from the correspondences and epipolar constraints of Eq. 2.10 alone [59]. The essential matrix may then be decomposed using Singular Value Decomposition (SVD) to determine the relative rotation and translation [59]. Similar to the essential matrix is the *fundamental matrix*, which describes the epipolar geometry when calibration is unknown. The relationship between the essential and fundamental relationships is described by the intrinsic camera calibration matrices: $F = K_1^{-\top} E K_0^{-1}$. If calibration is unknown and only the 2D-2D pixel correspondences are avilable then the fundamental matrix may be computed from seven or more correspondences [33, 32].

2.2.2 Robust Estimation With RANSAC

When estimating the relative pose between two images with the five-point [59] or seven-point algorithms [33], outliers in feature correspondences and noise in the estimation process can deteriorate the quality of the reconstruction process. To overcome these sources of error, two techniques are employed. First, Random Sample Consensus

(RANSAC) [25] estimation is used to determine camera geometry. RANSAC is a robust estimation procedure used to compute a model given data contaminated from outliers. Hypotheses are estimated from random subsets of the input data, and each data point is labelled an "inlier" or and "outlier" depending on if the data point agrees with the hypothesis (*e.g.*, if the residual error is sufficiently small). Thus, the probability of computing a high-quality model in the presence of outliers is approximately reduced to the probability of randomly sampling an all-inlier subset. If the ratio of inliers to outliers in the data is ε, and n points are required to estimate a model, then $1-\varepsilon^n$ is the probability that at least one of the n points chosen is an outlier (implying that a bad hypothesis will result). After choosing k random subsets, the probability that at least one of the k subsamples contains only inliers is

$$p = 1 - (1 - \varepsilon^n)^k. \tag{2.11}$$

Conversely, if we wish to determine the number of samples required in order to compute a high quality model with a given confidence (*e.g.*, 99% confidence), we can use the equation above to solve for k:

$$k = \frac{log(1-p)}{log(1-w^n)}. \tag{2.12}$$

Figure 2.4 demonstrates how the number of required samples increases as the number of points used to compute a model, n, increase. The number of required iterations increases exponentially as n increases and, as a result, n is typically set to the minimal number of data points required to compute a hypothesis. RANSAC may be applied to any model estimation procedure to robustly estimate a model hypothesis and has been shown to produce high quality results efficiently in practice. This has created a great interest in creating so-called "minimal solvers" in computer vision that compute

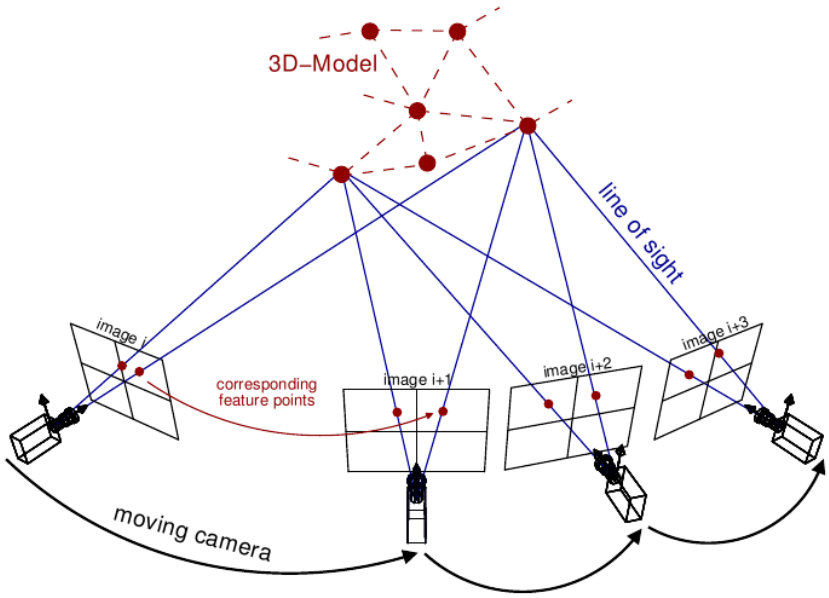

Figure 2.5: Incremental SfM begins with two cameras then adds new camera observations and 3D points one at a time to gradually build a 3D model.

hypotheses efficiently from a minimal sample [33, 59, 34, 45, 47, 46]. Similarly, many RANSAC variants have been proposed that optimize the sampling strategy, residual evalaution, and model quality [16] [19] [17] [18] [26] [66] [83].

2.2.3 Incremental SfM

In incremental SfM, a 3D reconstruction is gradually grown by beginning with 2 cameras and adding additional cameras and 3D points sequentially to determine the camera poses and positions of 3D points in a global reference frame. To begin this process, we choose a pair of images as our initial seed for the reconstruction. This pair is typically chosen as the pair that contains the largest number of correspondences with care to avoid pairs observing degenerate structures such as planes [73]. After the initial 3D reconstruction is created, new images are successively added to grow the reconstruction. The camera pose of new cameras is determined from the 2D-3D correspondences between

image points in the new image and existing 3D points in the current reconstruction. New 3D points are triangulated based on observations from the new camera. This process is repeated, adding the images that observe the highest number of 3D points currently in the scene and triangulating new 3D points until all possible images have been added to the reconstruction.

To handle outliers in feature correspondences and noise in the geometry estimation process, two techniques are employed. First, RANSAC is used to robustly compute the poses of cameras from 2D-3D correspondences using the P3P algorithm [45]. This allows for efficient and accurate computation of the camera pose in the presence of outliers. Still, small inaccuracies exist in the pose estimation process due to noise. To avoid accumulating this error, *bundle adjustment* is used to refine the camera poses and 3D point locations[85]. This nonlinear optimization procedure refines the 3D reconstruction by minimizing the sum of squared *reprojection errors* to minimize the pixel-distance between 2D observations and the projection of their corresponding 3D point into the image:

$$\underset{K_i,R_i,t_i,X_j}{\arg\min} \sum_i \sum_j \delta_{i,j} ||x_j^i - \Pi(K_i(R_i X_j + t_j))||^2 \tag{2.13}$$

where Π is the projection function $\Pi(x, y, z) = \Pi(x/z, y/z)$ and $\delta_{i,j}$ is a binary value equal to 1 when camera i observes point j and 0 when it does not. $x_{i,j}$ and X_j are 2D image observations and 3D points in the scene respectively.

This joint refinement of camera and point parameters is extremely expensive, especially as the size of the scene grows. Much effort has gone towards improving the efficiency and quality of bundle adjustment through various minimization strategies [23] [53] [85] [84] [5], preconditioners [48] [37], and parallel algorithms [90], and useful open-source software such as SBA [54] and Ceres Solver [4] enable bundle adjustment to be

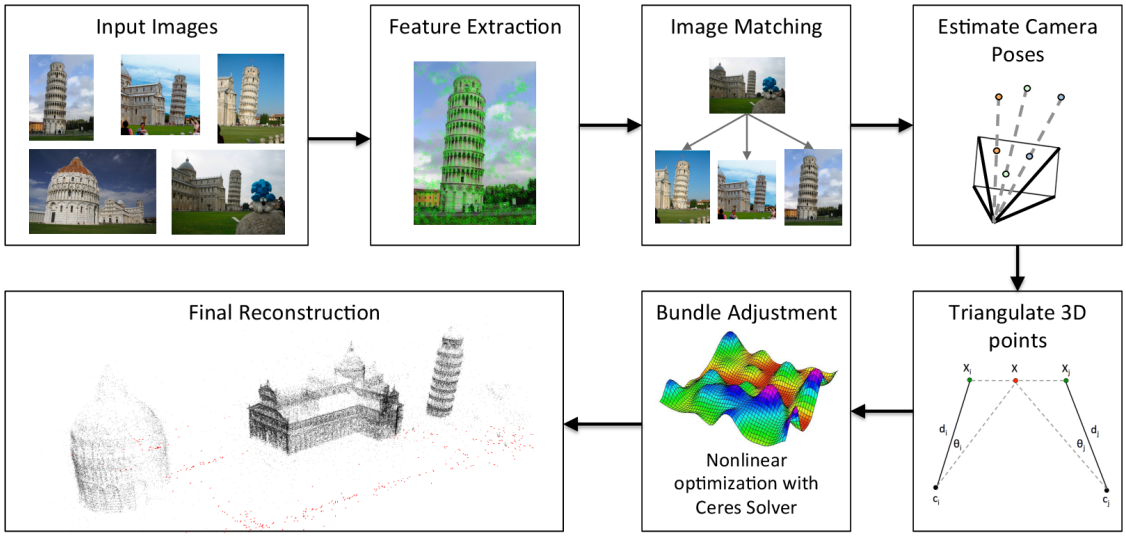

Figure 2.6: In contrast to incremental SfM, global SfM solves for all camera poses simultaneously by first determining the camera orientations, then the camera positions. Global SfM is more efficient and scalable than incremental SfM because the expensive nonlinear optimization of bundle adjustment is only required once.

performed efficiently. In incremental SfM, bundle adjustment is repeatedly applied as the model grows to overcome the effects of noise during the reconstruction process. While this typically results in high-quality reconstructions, the repeated use of bundle adjustment is the main source of inefficiency in incremental SfM and limits the scale of the 3D reconstructions that may be computed with incremental methods without the use of computing clusters [2]. Alternatively, global SfM methods only require a single bundle adjustment at the end of the model estimation resulting in a significant improvement in efficiency over incremental methods.

2.2.4 Global SfM

While incremental SfM can compute small to medium-sized reconstructions (several hundred images or fewer) efficienctly, the repeated use of bundle adjustment limits the

scalability of such approaches. Further, the sequential nature of incremental methods inherently weighs images added earlier in the reconstruction process more heavily [74]. These images are not inherently "more important" than images added later, and more natural approach would weigh all images equally.

Global SfM methods overcome these limitations by considering all cameras at the same time and computing the pose of all cameras simultaneously with motion averaging algorithms [29] [30]. As such, all images are weighted equally and order does not affect the final outcome. To estimate camera poses, the relative poses between pairs of images are utilized to compute estimates of the poses in a global reference frame (*cf*. Figure 2.6). Only images pairs that have a sufficient number of inliers after epipolar geometry estimation are considered. First, the rotations of camera are estimated from the relative rotations by noting that:

$$R_{i,j} = R_j R_i^\top \tag{2.14}$$

where $R_{i,j}$ is the relative rotation from camera i to camera j, and R_i, R_j are the world-to-camera rotations of camera i and j (*i.e.*, the extrinsic camera parameters for rotation). Due to noise and outliers, Eq. (2.14) generally does not hold; however, we may use this equation to form a cost function in terms of the relative and absolute rotations. When all relative rotation constraints are considered simultaneously, the minimization of this cost function produces estimates for the camera rotation of all cameras in a global coordinate system.

$$\arg\min_{R_i} = \sum_{i,j} R_{i,j} R_i - R_j. \tag{2.15}$$

This constraint may be used to compute camera rotations using efficient solvers [6, 14].

After camera rotations have been computed, camera positions may be computed using

a similar constraint to Eq. 2.14 for relative translations:

$$t_{i,j} = R_i \frac{c_j - c_i}{||c_j - c_i||} \tag{2.16}$$

where $t_{i,j}$ is the unit-norm relative translation between cameras i and j, c_i and c_j are the camera positions in a global coordinate system, and R_i is the rotation of camera i computed in the previous step. In contrast to the rotation estimation step, Eq. 2.16 is nonlinear and a globally optimal solution is difficult to compute. Several position estimation methods exist to compute the position from linear cost function [29] [60] [57] [38] [22] or with robust estimation techniques [88] [20] [61].

After computing the camera positions in a global coordinate system, the 3D points are triangulated and bundle adjustment is performed to refine the 3D points and camera poses. Notice that triangulation and bundle adjustment are performed only once, leading to dramatic efficiency increases for global SfM in large-scale datasets. The accuracy of these methods, particularly the position estimation methods, is worse than incremental SfM because no image observations are used to compute camera poses. This is in direct contrast with incremental methods that repeatedly refine camera poses with bundle adjustment throughout the reconstruction process to minimize reprojection error of the image observations. Indeed, there is a tradeoff between the accuracy and robustness of incremental SfM methods and the scalability of the global methods. One focus of this thesis is to remove the gap by designing new robust, accurate, and scalable SfM methods.

Part I

Modeling the Distributed Camera

Chapter 3

A Full Generalization of the Relative Pose Problem

When two images observe the same scene, the *relative pose problem* aims to compute the relative geometry between two cameras from image correspondences alone. This relative pose is defined by the epipolar geometry between the two images (see Chapter 2.2.1 for more details). The relative pose is a core building block for many real-time SLAM system such as PTAM [42], and is the first step in incremental SfM [73]. Given the relative camera pose, 3D points may be triangulated to obtain a full 3D reconstruction of cameras and structure.

In a projective setting (*i.e.*, calibration is not known), the Fundamental Matrix is a 3×3 matrix that encapsulates the relative pose between two cameras such that:

$$x'^{\top} F x = 0 \tag{3.1}$$

where x is in the first view and x' is in the second view. Similarly, when calibration is known the 3×3 Essential Matrix E describes the relative camera geometry such that:

$$x'^{\top} E x = 0. \tag{3.2}$$

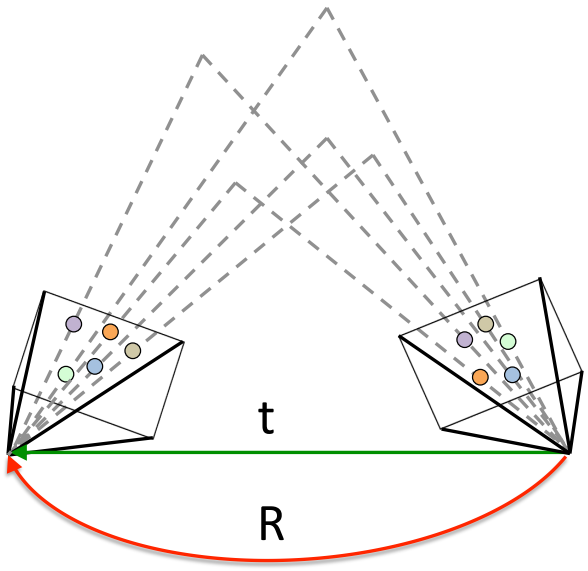

Figure 3.1: The relative camera pose problem determines a camera's position and orientation with respect to a second camera using 5 image-to-image correspondences.

The relative pose may be recovered from the essential matrix by noting the $E = [t]_x R$ where t is the relative translation (up to scale) and R is the relative rotation between the two cameras.

In this chapter, we focus on the calibrated relative pose problem to directly recover relative translations and rotations between cameras. This particular problem is a fundamental building block for SfM, as it is the starting point for incremental SfM algorithms and is the input for global SfM algorithms. We extend the standard relative pose problem first to multi-camera systems (*i.e.*, generalized cameras) then additionally incorporate scale changes. Incorporating scale changes between generalized cameras is equivalent to recovering a 7 degrees-of-freedom (d.o.f.) similarity transformation and allows for a much broader use of generalized cameras. In particular, similarity transformations can be used for loop closure in SLAM (where scale drift occurs) and for merging multiple Structure-from-Motion (SfM) reconstructions when the scale between the reconstructions

is unknown. This problem arises frequently because scale cannot be explicitly recovered from images alone without metric calibration, so developing accurate, efficient, and robust methods to solve this problem is of great importance. We show that this generalizing the relative pose problem in this way leads to a single solution method for all relative pose problems and enables efficient model-merging.

3.1 A Quadratic Eigenvalue Formulation

We will first consider the problem of determining the relative pose between two cameras, and will then extend this formulation to the general case. In order to eventually derive a general formulation for the relative pose problem, we do not consider Essential/Fundamental Matrices but instead work directly with the relative motion. Thus, computing the relative pose between two cameras requires solving for the relative rotation and translation that satisfies:

$$x'^{\top}[t]_x R x = 0. \tag{3.3}$$

Using the scalar triple product rule, this constraint may be rewritten as:

$$(\overline{x_i} \times R\overline{x_i}') \cdot t = 0 \tag{3.4}$$

If we collect this constraint for all correspondences, we have:

$$m_i(R)^{\top} \cdot t = 0 \qquad m_i(R) = Rx_i \times x_i',\ i = 1 \dots n \tag{3.5}$$

$$[m_1(R) \dots m_n(R)]^{\top} \cdot t = 0 \tag{3.6}$$

$$M(R)^{\top} \cdot t = 0 \tag{3.7}$$

where $M(R)$ is a $3 \times n$ matrix that is a function of the unknown rotation R and known pixel observations. In the minimal case, only 5 correspondences are needed to estimate the relative pose. The expression in Eq. (3.7) is very powerful, as it indicates that the solution for the relative translation lies in the null space of M. In other words, we know that the expression M is rank deficient and, specifically, is rank 2. We use this knowledge to help solve for the unknown rotation.

To solve for the relative pose, let us consider the quaternion rotation parameterization $q = (x,\ y,\ z,\ \alpha)^\top$ such that the rotation matrix

$$R = 2(vv^\top + \alpha[v]_\times) + (\alpha^2 - 1)I, \tag{3.8}$$

where $v = (x,\ y,\ z)^\top$ and $[v]_x$ is the skew-symmetric cross product matrix of v. Thus, M is quadratic in the quaternion parameters and the generalized epipolar constraint of Eq. (3.7) is a 4-parameter Quadratic Eigenvalue Problem (QEP) of the form:

$$\begin{aligned}(&M_0x^2 + M_1y^2 + M_2z^2 + M_3\alpha^2 \\ &+ M_4xy + M_5xz + M_6x\alpha + M_7yz + M_8y\alpha + M_9z\alpha \\ &+ M_{10}x + M_{11}y + M_{12}z + M_{13}\alpha + M_{14}) \cdot t = 0\end{aligned} \tag{3.9}$$

No methods currently exist to directly solve a 4-parameter QEP and it should be noted that a non-iterative solution to Multiparameter Eigenvalue Problems with more than two parameters is an open problem in mathematics. However, an iterative optimization similar to [43] may be used to minimize the smallest eigenvalue of M and determine the unknowns if a good initialization is available.

Since the 4-parameter QEP of Eq. (3.9) is intractable, we instead solve the problem in two steps to make the problem tractable. First, we align the vertical directions of the

cameras. This may be done by utilizing IMU data, detecting vertical vanishing points, or with the robust alignment method described in [80]. Then we utilize the knowledge of the vertical direction to formulate the relative pose problem as a Quadratic Eigenvalue Problem which is simple to construct and efficient to solve.

3.1.1 Solving the Quadratic Eigenvalue Problem

By assuming the vertical direction between the two cameras has been aligned we remove 2 d.o.f. from the unknown rotation. The only remaining d.o.f. of the relative rotation corresponds to determining an angle of rotation about the vertical axis. By removing 2 d.o.f. from the problem, it is clear to see that the minimum number of correspondences required to solve for the relative pose is reduced from 5 correspondences to 3.

In terms of the quaternion parameterization of Eq. 3.8, $v = [xyz]^\top$ corresponds to the vertical axis, and so R is quadratic in terms of only the unknown variable α. Thus, our 4-parameter QEP of Eq. 3.9 now becomes a 1-parameters QEP of the form:

$$(\alpha^2 A + \alpha B + C) \cdot t = 0, \tag{3.10}$$

where A, B, and C are 3×3 matrices formed from matrix M in Eq. (3.7). We now have a standard 1-parameter QEP which has been thoroughly examined in linear algebra [82]. To solve this QEP, we first convert it to a Generalized Eigenvalue Problem of the form:

$$\begin{bmatrix} B & C \\ -I & 0 \end{bmatrix} z = s \begin{bmatrix} -A & 0 \\ 0 & -I \end{bmatrix} z, \tag{3.11}$$

where $z = \begin{bmatrix} \alpha t^\top & t^\top \end{bmatrix}^\top$ is the eigenvector and s is the eigenvalue. This can be converted

to a standard eigenvalue problem by inverting the right-hand matrix of Eq. (3.11). The inverse is particularly simple and efficient in this case:

$$\begin{bmatrix} -A & 0 \\ 0 & -I \end{bmatrix}^{-1} = \begin{bmatrix} -A^{-1} & 0 \\ 0 & -I \end{bmatrix}.$$

The Generalized Eigenvalue Problem of Eq. (3.11) may now be reduced to a standard eigenvalue problem,

$$\begin{bmatrix} -A^{-1}B & -A^{-1}C \\ I & 0 \end{bmatrix} z = sz,$$

which can be solved with standard methods. The solution to this produces 6 candidate solutions where the eigenvalues correspond to α and the translation and scale may be extracted from the eigenvector z. We may eliminate some of the candidate solutions by only considering real eigenvalues and the eigenvectors where the first 5 entries are equal to the last 5 entries scaled by α to ensure our solution is consistent with the construction of vector z. In general, 2 of the 6 solutions correspond to imaginary eigenvalues and can be immediately removed from consideration.

3.1.2 A Closed Form Solution

An alternative method for solving Eq. (3.10) arises by examining the determinant. Since M from Eq. (3.22) will be rank-deficient in non-degenerate cases, so it must hold that:

$$det(\alpha^2 A + \alpha B + C) = 0. \tag{3.12}$$

This leads to a degree 10 univariate polynomial in α such that the roots correspond to valid solutions to α. Further, it can be shown that this polynomial is always divisible by

$\alpha^2 + 1$, leading to at most 4 real solutions. This result also verify that our QEP method will have at most 4 real solutions since the roots of this polynomial correspond to the eigenvalues of our QEP. In practice this polynomial is often ill-conditioned and solutions may be unstable. The loss in numerical precision and accuracy is not worth the 10-20% speed increase so in practice we only consider the QEP method.

3.2 Generalization to 7 d.o.f.

A generalization of the relative pose problem is to compute the relative pose between two sets of multiple cameras. Each set of multiple cameras may be described by the generalized camera model which allows a set of image rays that do not necessarily have the same ray origin to be represented in a uniform expression. Generalized cameras are extremely useful for many practical applications such as omni-directional camera systems and vehicle-mounted multi-camera systems. Solutions exist for computing relative pose between generalized cameras [43, 50, 76]; however, these methods require that the internal scale of the multi-camera system (*i.e.*, the distance between all camera centers within the multi-camera system) is known. This limits the use of generalized cameras to cases where scale calibration can be easily captured. In this section, we provide a further generalization of the relative pose problem and remove the requirement of known scale to solve a new problem: the generalized relative pose and scale problem.

The generalized relative pose and scale problem is a direct generalization of the generalized relative pose problem. The generalized relative pose problem uses ray correspondences to compute the rotation and translation that will transform one set of rays so that they intersect with the second set of rays. Let $\overline{x_i}$ and $\overline{x_i}'$ be corresponding unit vectors that intersect in 3D space with ray origins c_i and c_i'. These rays can be represented in

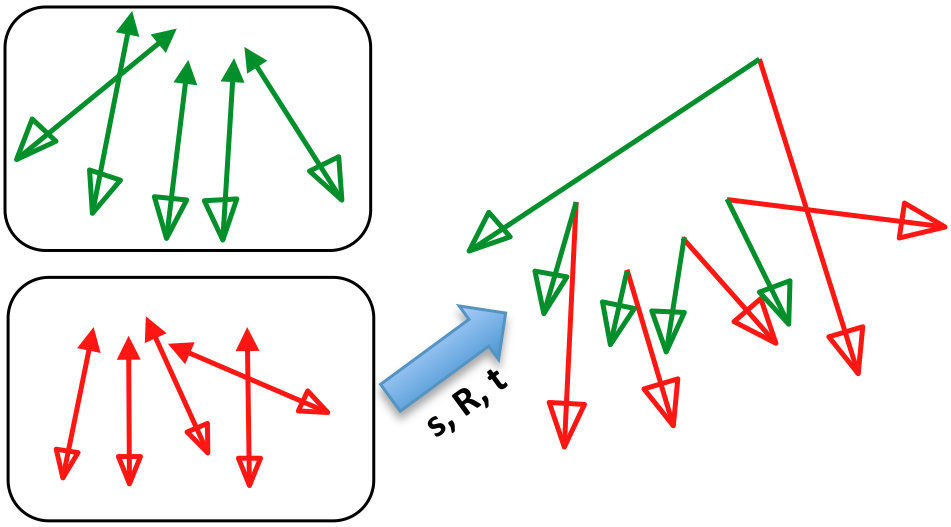

Figure 3.2: We present a method to solve the generalized relative pose and scale problem. We first align the generalized cameras to a common vertical direction then use image rays obtained from 5 2D-2D correspondences to solve for the remaining degrees of freedom. Solving this problem is equivalent to computing a similarity transformation

Plücker coordinates [64] such that:

$$l_i = \begin{pmatrix} \overline{x}_i \\ c_i \times \overline{x}_i \end{pmatrix} \text{ and } l'_i = \begin{pmatrix} \overline{x}_i{}' \\ c'_i \times \overline{x}_i{}' \end{pmatrix}. \tag{3.13}$$

The generalized epipolar constraint [63] that describes the intersection of two Plücker coordinates may then be written as:

$$(\overline{x}_i \times R\overline{x}_i{}')^\top t + \overline{x}_i{}^\top([c_i]_\times R - R[c'_i]_\times)\overline{x}_i{}' = 0, \tag{3.14}$$

where R and t are the rotation and translation that transform $\overline{x}_i{}'$ and c'_i such that the ray correspondences intersect in 3D space. This problem has been solved previously with minimal [76], linear [50], and nonlinear approaches [43]. Further, the generalized relative

pose problem can be represented by our QEP formulation:

$$m_i(R)^\top \cdot t' = 0, \text{where} \tag{3.15}$$

$$m_i(R) = \begin{pmatrix} \overline{x_i} \times R\overline{x_i}' \\ \overline{x_i}^\top([c_i]_\times R - R[c_i']_\times)\overline{x_i}' \end{pmatrix} \text{ and } t' = \begin{pmatrix} t \\ 1 \end{pmatrix} \tag{3.16}$$

$$M(R)^\top \tilde{t} = (m_1(R) \quad \dots \quad m_n(R))^\top \tilde{t} = 0. \tag{3.17}$$

The rank-deficient matrix $M(R)$ is a function of R and thus the QEP method described in Section 3.1 may be used to solve for the relative pose between the generalized cameras. A limitation of this formulation is that it assumes the scale between the two generalized cameras has been reconciled yet in many cases the scale is not available or may be inherently ambiguous without metric calibration (*e.g.*, in SfM reconstructions). Thus, we are interested in additionally solving for the unknown scale transformation between the two generalized camera.

To solve the generalized relative pose and scale problem we must additionally recover the unknown scale s that stretches the ray origins c_i'. Thus, the generalized epipolar constraint becomes:

$$(\overline{x_i} \times R\overline{x_i}')^\top t + \overline{x_i}^\top([c_i]_\times R - Rs[c_i']_\times)\overline{x_i}' = 0 \tag{3.18}$$

$$(\overline{x_i} \times R\overline{x_i}')^\top t - s\overline{x_i}^\top R[c_i']_\times \overline{x_i}' + \overline{x_i}^\top [c_i]_\times R\overline{x_i}' = 0. \tag{3.19}$$

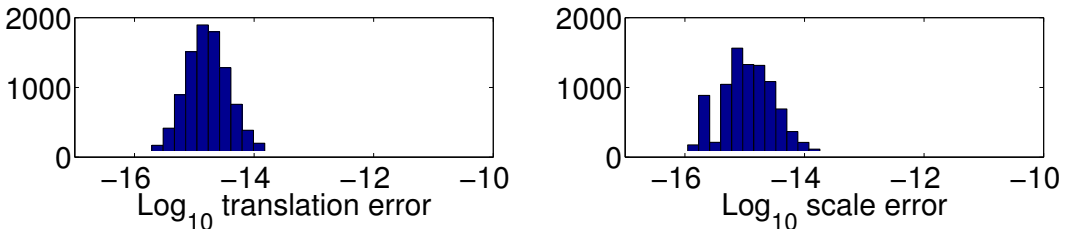

Figure 3.3: We measured the numerical stability of our algorithm with zero pixel noise and a perfect known axis of rotation. The translation and scale errors are very small, and the rotation error cannot be displayed because it was within the machine precision.

As before, this equation may be rewritten as:

$$m_i(R)^\top \cdot \tilde{t} = 0, \text{where} \tag{3.20}$$

$$m_i(R) = \begin{pmatrix} \overline{x_i} \times R\overline{x_i}' \\ -\overline{x_i}^\top R[c_i']_\times \overline{x_i}' \\ \overline{x_i}^\top [c_i]_\times R\overline{x_i}') \end{pmatrix} \text{ and } \tilde{t} = \begin{pmatrix} t \\ s \\ 1 \end{pmatrix} \tag{3.21}$$

$$M(R)^\top \tilde{t} = (m_1(R) \quad \dots \quad m_n(R))^\top \tilde{t} = 0. \tag{3.22}$$

The generalized relative pose and scale problem has 7 d.o.f. and thus requires 7 correspondences in the minimal case. Once again, we may utilize knowledge of the vertical direction to reduce the 4-parameter QEP to a 1-parameter QEP. This allows us to solve the generalized relative pose and scale problem from 5 image-to-image correspondences.

3.3 Experimental Evaluation

3.3.1 Numerical stability

We tested the numerical stability of our QEP method over 10^5 random trials. We generated random camera configurations that placed cameras (*i.e.*, ray origins) in the

cube $[-1,1]\times[-1,1]\times[-1,1]$ around the origin. 3D points were randomly placed in the cube $[-1,1]\times[-1,1]\times[4,6]$ and ray directions were computed as unit vectors from camera origins to 3D points. Correspondences were computed from image rays that observed the same 3D points. An identity similarity transformation was used (*i.e.*, $R = I$, $t = 0$, $s = 1$). For each trial, we computed solutions using the minimal 5 correspondences. We calculated the angular rotation error, the translation error, and the scale error for each trial, and plot the results in Figure 3.3. The errors are very stable, with 99% of all errors less than 10^{-12}.

3.3.2 Image noise experiment

We performed experiments on synthetic data to determine the effect of image noise on our algorithm. We compared our algorithm to three alternative algorithms: the gDLS algorithm [79], the gP+s algorithm [87], and the Absolute Orientation algorithm [86].

For our synthetic setup we generated two generalized cameras that each consist of 5 cameras randomly placed in the $2\times2\times2$ cube centered at the origin. 3D points were then randomly generated with a mean distance of 5 units from the origin, and correspondences were established as rays that observed the same 3D points such that each camera observes a single 3D point. We then applied a similarity transformation with a random rotation, a translation in a random direction with a random baseline in the range of $[0.1,\ 100]$, and a random scale in the range of $[0.1,\ 100]$ to the second generalized camera. Image noise is added to the second generalized camera and the similarity transformation is estimated. We report the angular rotation error, absolute translation error, and the normalized scale error $|s-\hat{s}|/s$.

For all synthetic experiments we used the ground truth vertical direction and added 0.5 degrees of Gaussian noise to simulate the real accuracy of vertical direction esti-

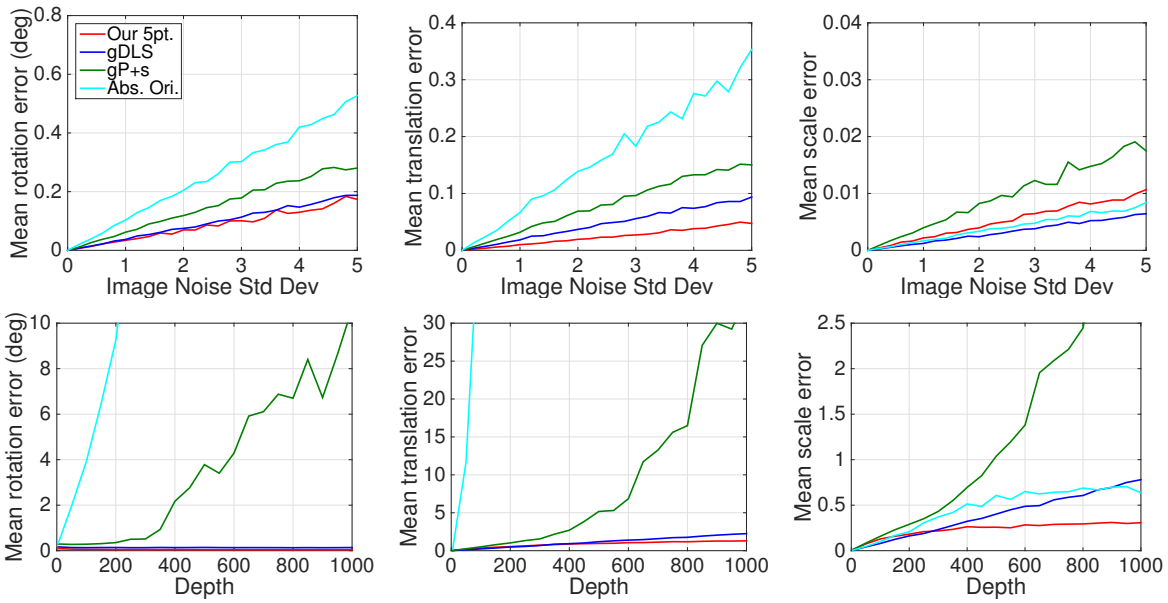

Figure 3.4: We measured the error in the computed similarity transformation as the amount pixel noise increased and plot the mean rotation, translation and scale error. All cameras were randomly generated within a $2 \times 2 \times 2$ cube centered at the origin. **Top row:** we generated random 3d points with an average depth of 5 units away from the orign. Our algorithm is the most accurate at computing the rotation and translation but is not as accurate at computing scale, however, the scale errors are very small for all algorithms. **Bottom row:** we kept the image noise at 1.0 pixels standard deviation while increasing the average depth of hte 3D points used to establish correspondences. Our algorithm is least affected by the change in scene depth meaning that it is robust to uncertainty in 3D point positions.

mation for our algorithm. For the Absolute Orientation algorithm, we created 3D-3D matches by triangulating 3D points in the second generalized camera from the noisy image rays and used these 3D points to establish correspondences. Additionally, we used 5 correspondences for each algorithm for a fair comparison.

Using the setup described, we ran 1000 trials testing the accuracy of each algorithm as increasing levels of image pixel noise were added (Figure 3.4 top). Scenes were randomly generated for each trial, and all algorithms used the same scene configuration for a given trial. Our algorithm performed best at estimating the rotation and translation of the similarity transformation but is less accurate than the gDLS and Absolute Orientation

algorithms for estimating scale. It should noted that the scale errors are very small for all algorithms. Our algorithm is robust to image noise because ray intersection in 3D space is a very tight constraint that is independent of the depth of the 3D point.

3.3.3 Scene depth experiment

In SLAM and SfM it is common to have 3D points with large and varying scene depth. It is especially important in the case of urban and large-scale SfM to be robust to large scene depths when computing a similarity transformation to align models. To examine our algorithm's robustness to scene depth, we ran an experiment using the same setup as above while increasing the mean scene depth from 5 units to 200 units. We used an image noise of 1 pixel for all depth levels and executed 1000 trials at each depth level. The results of our experiment are shown in the bottom row of Figure 3.4. It is clear to see that our algorithm is least affected by scene depth. The Absolute Orientation and gP+s algorithms completely degrade as the scene depth increases. The gDLS algorithm has comparable depth robustness to our algorithm in terms of the rotation and translation but is not as accurate at computing scale.

Conceptually, our algorithm has an advantage over gDLS [79], gP+s [87], and the Absolute Orientation algorithm [86] because it does not use 3D points and thus is not subject to uncertainty in the 3D position. It is well known that the uncertainty of a triangulated 3D point increases as the depth of the point relative to the baseline of the cameras observing it increases. Therefore, our algorithm should produce more accurate similarity transformations as the scene depth increase. Indeed, the results of this experiment support this notion.

3.3.4 IMU noise experiment

We performed experiments on synthetic data to determine how the accuracy of the estimated vertical direction affects our algorithm. To simulate noise in the estimated vertical direction we added gaussian noise to a synthetic IMU ranging from 0 to 1 degree of standard deviation.

Using the same scene setup as the image noise experiment, we ran 1000 trials testing the similarity transformation accuracy as increasing levels of IMU noise were added (Figure 3.5). Standard mobile devices have less than 0.5 degree of IMU noise with high quality sensors often having less than 0.01 degrees of noise. Our algorithm demonstrates good accuracy in the presence of IMU noise within this range, verifying its robustness to potentially inaccurate vertical direction estimations.

3.3.5 Time Complexity

A major benefit of our method is that the QEP solution is simple to construct and very efficient. The most costly operations involved in our method are inversion of a 5×5 matrix and computing the eigenvectors and eigenvalues of a 10×10 matrix. Both of these operations are highly efficient on small matrices in standard linear algebra packages. Over 10,000 trials our algorithm ran with a mean execution time of 44μs. In comparison, the gDLS [79] method had a mean execution time of 606μs and the gP+s [87] method had a mean execution time of 118 μs. All timing experiments were run on a 2011 Macbook Pro with a 2GHz Intel Core i7 processor. While the Absolute Orientation algorithm is more efficient at 3μs, it is not as accurate or as robust to image noise and depth variance as our algorithm (*cf*. Figure 3.4). Our algorithm has comparable accuracy to gDLS in the presence of image noise and is more robust to depth variance, yet it has a speedup of over $10\times$. This makes our algorithm more desirable for real-time use in a RANSAC

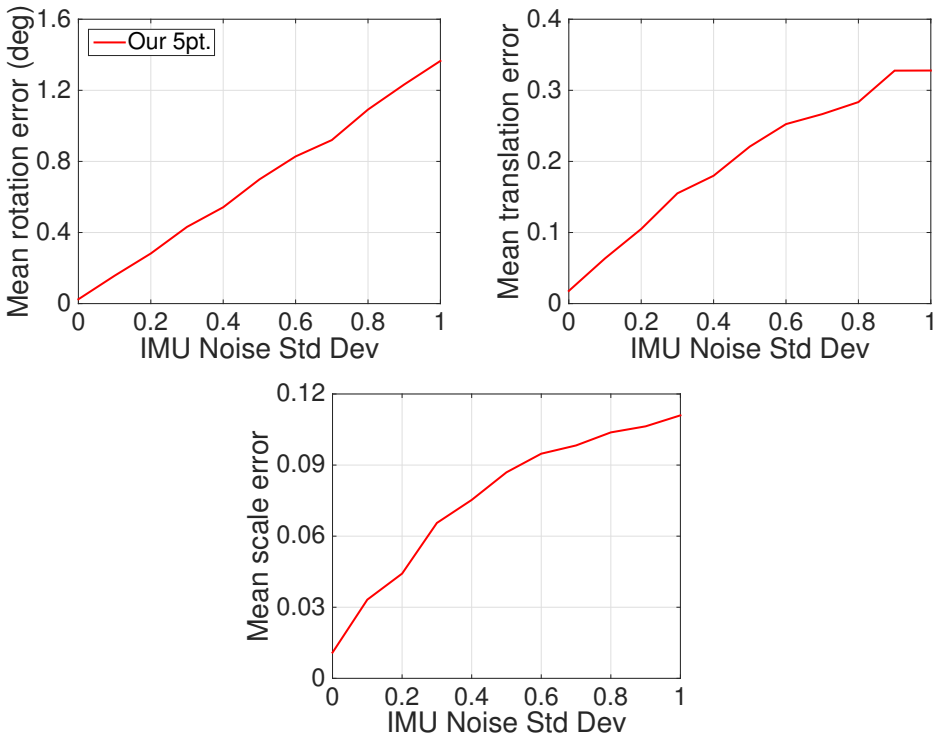

Figure 3.5: Using the same scene configuration as the image noise experiment, we measured the similarity transformation error as noise was added to the synthetic IMU to perturb the vertical direction. We only show our algorithm since it is the only one that depends on knowledge of the vertical direction. We used 1 pixel of image noise for all experiments. For levels of IMU noise expected on mobile devices (less than 0.5 degrees) our algorithm still maintains good accuracy, demonstrating robustness to noise in the vertical direction estimation.

scheme because of speed gains that will be realized.

3.3.6 Real-data experiments

Our method's robustness to 3D point and depth variance makes it well-suited for real-world applications. We tested the performance of our solver using the SLAM dataset from [87] that has highly accurate ground truth poses obtained with an ART-2 optical tracker for measuring the error of our similarity transformation registration method. Example images from this dataset are provided in Figure 3.6. For our experiment, we created an SfM reconstruction (using the ground truth poses) from one image sequence

Table 3.1: Average position error in centimeters for aligning a SLAM sequence to a pre-existing SfM reconstruction. An ART-2 tracker was used to provide highly accurate ground truth measurements for error analysis. Camera positions were computed using the respective similarity transformations and the mean camera position error of each sequence is listed below. Our method is has comparable or better accuracy than the state-of-the-art method, gDLS, but does not require any 3D points.

Sequence	# Images	Abs. Ori. [86]	gP+s[87]	gDLS [79]	Our 5 pt.
office1	9	6.37	6.12	**3.97**	4.30
office2	9	8.09	9.32	5.89	**4.17**
office3	33	8.29	6.78	6.08	**5.10**
office4	9	4.76	4.00	3.81	**2.61**
office5	15	3.63	4.75	**3.39**	3.41
office6	24	5.15	5.91	**4.51**	4.81
office7	9	6.33	7.07	4.65	**4.06**
office8	11	4.72	4.59	**2.85**	3.12
office9	7	8.41	6.65	3.19	**2.62**
office10	23	5.88	5.88	4.94	**3.55**
office11	58	5.19	6.74	**4.77**	5.03
office12	67	5.53	4.86	4.81	**4.12**

to use as our reference image sequence and point cloud. We then run 12 image sequences through a keyframe-based SLAM system to obtain a local tracking sequence that can be registered with respect to the reference sequence with a similarity transformation (see Figure 3.7). We then compute a similarity transformation in the following manner:

Our 5 pt.: 2D-2D feature correspondences are established between the reference and query image sequences using an approximate nearest neighbor search (ANN), and the vertical directions are aligned using ground plane detection and computing the normal. These correspondences are then used in a RANSAC loop with the 5 pt. method described in this paper to determine a similarity transformation.

gDLS: We obtain 2D-3D correspondences with an ANN search between the 3D points in the point cloud generated by the reference sequence and the 2D image features in the query sequences. These correspondences are then used in a RANSAC loop using the

minimal number of 4 correspondences with the gDLS algorithm of Sweeney *et al.* [79].

gP+s: We obtain 2D-3D correspondences in the same way as the gDLS method and use these correspondences in a RANSAC loop with the algorithm of Ventura *et al.* [87] to estimate the similarity transformation. This method requires 4 correspondences in the minimal case.

Absolute Orientation: The absolute orientation method of Umeyama [25] is used to align the 3D points from the reference point cloud to 3D points triangulated from 2D correspondences in the query point cloud. Correspondences are determined from an ANN search of the mean descriptor of the triangulated point and the 3D points in the reference point cloud. We use 4 correspondences for this method.

After applying the computed similarity transformation directly from RANSAC (*i.e.*, no refinement is performed), we compute the average position error of all keyframes with respect to the ground truth data. We report the mean position error of all keyframes in the image sequence (in centimeters) over 1000 trials in Table 3.1. Our method performs better than all other methods in most of the scenes. The globally optimal gDLS algorithm [79] is the only method that is competitive with our algorithm. We expect that our algorithm will perform even better for large-scale SfM applications. However, acquiring ground truth datasets for large-scale SfM is difficult and we leave the incorporation and evaluation of our algorithm into a large-scale hierarchical SfM pipeline for future work.

3.4 Discussion

In this chapter, we have examined a fundamental computer vision problem, the relative pose problem, and viewed it in the context of a distributed camera. Using the standard single-camera relative pose constraints, we provide a mathematical generalization that naturally extends to multiple cameras and scale changes thus making it useful

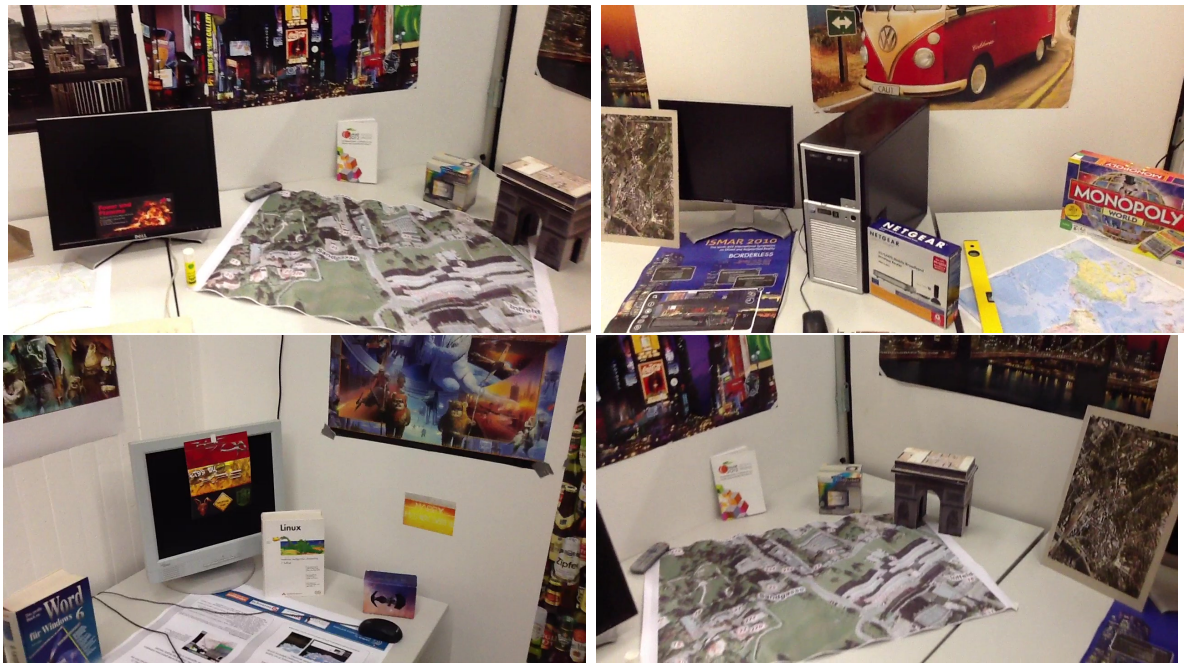

Figure 3.6: Example images from our real data experiments. The images created a SLAM sequence that was then aligned to a reference sequence with our method to estimate a similarity transformation.

for determining the relative pose between two distributed cameras. This generalization leads to a new problem called the generalized relative pose and scale problem and to our knowledge we have provided the first solution to this problem. The generalized relative pose and scale problem is equivalent to estimating a 7 d.o.f. similarity transformation and so this work is useful for loop closure in visual odometry and merging SfM reconstructions. We showed that the standard generalized relative pose and scale problem leads to an intractable 4-parameter QEP and instead provide a two step solution to the problem where we first align the vertical directions of all cameras then reduce the problem to a 1-parameter QEP that can be solved with standard linear algebra. Our method is simple, efficient, and robust to image noise and scene depth. We show on synthetic and real data experiments that our method has comparable or better performance to alternative algorithms. In future work, we hope to remove the necessity for vertical alignment to

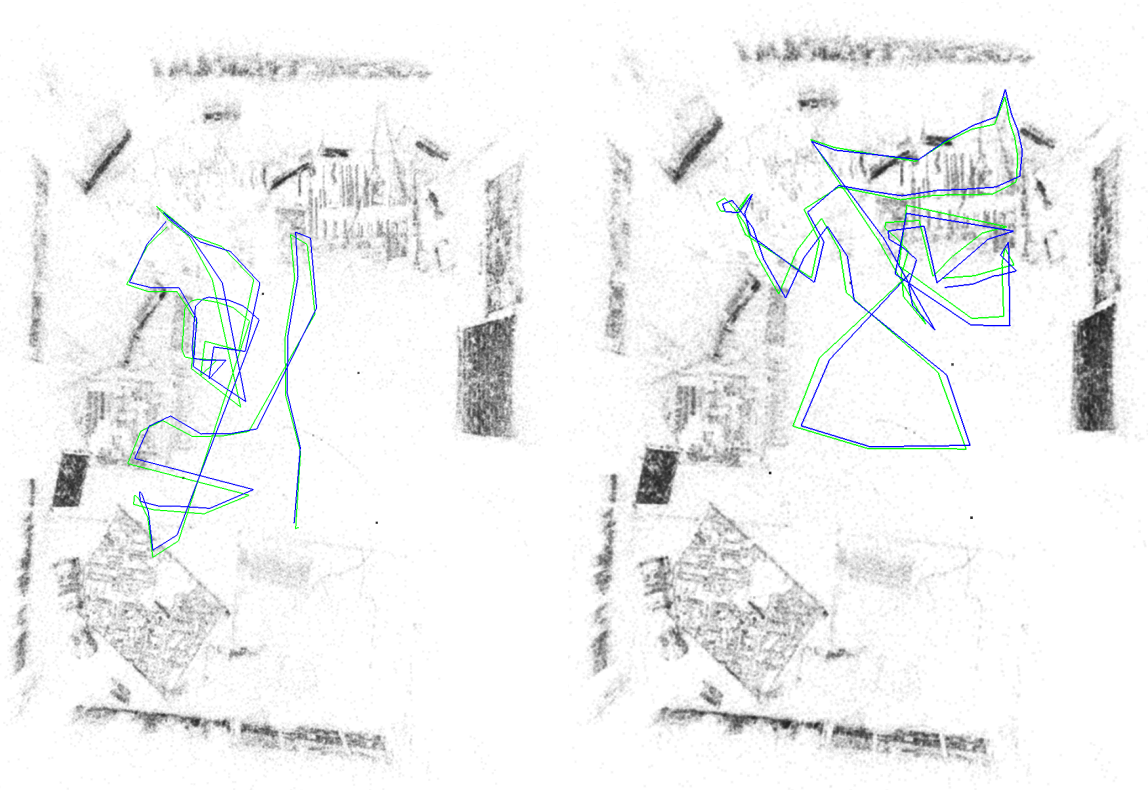

Figure 3.7: We compare our method with several alternative methods for computing similarity transformation using a dataset comprised of SLAM sequences that contain highly accurate ground truth poses. Each method is used to align 12 image sequences and the camera position errors are reported in Table 3.1. Green represents the ground truth SLAM sequence and blue SLAM sequence after applying the similarity transformation with our method in a RANSAC scheme.

allow additional flexibility to our algorithm, and would like to incorporate this method into a large scale multi-camera SfM pipeline where the scale of reconstructions may be ambiguous.

Chapter 4

A Full Generalization of the Absolute Pose Problem

The problem of determining camera position and orientation given a set of correspondences between 2D image observations and known 3D points is a fundamental problem in computer vision known as *the absolute camera pose problem* (*cf*. Figure 4.1 left) or the Perspective n-Point (PnP) problem. This problem has a wide range of applications in computer vision, including camera calibration, object tracking, simultaneous localization and mapping (SLAM), structure-from-motion (SfM), and augmented reality. In the minimal case, only three 2D-3D correspondences are required to compute the absolute camera pose [25, 34, 45]. These methods utilize the reprojection constraint such that 3D points X_i align with unit-norm pixel rays $\overline{x_i}$ when rotated and translated:

$$\alpha_i \overline{x_i} = RX_i + t, \qquad i = 1, 2, 3 \tag{4.1}$$

where R and t rotate and translate 3D points into the camera coordinate system. The scalar α_i stretches the unit-norm ray x_i such that $\alpha_i = ||RX_i + t||$. In order to determine the camera's pose, we would like to solve for the unknown rotation R and translation t

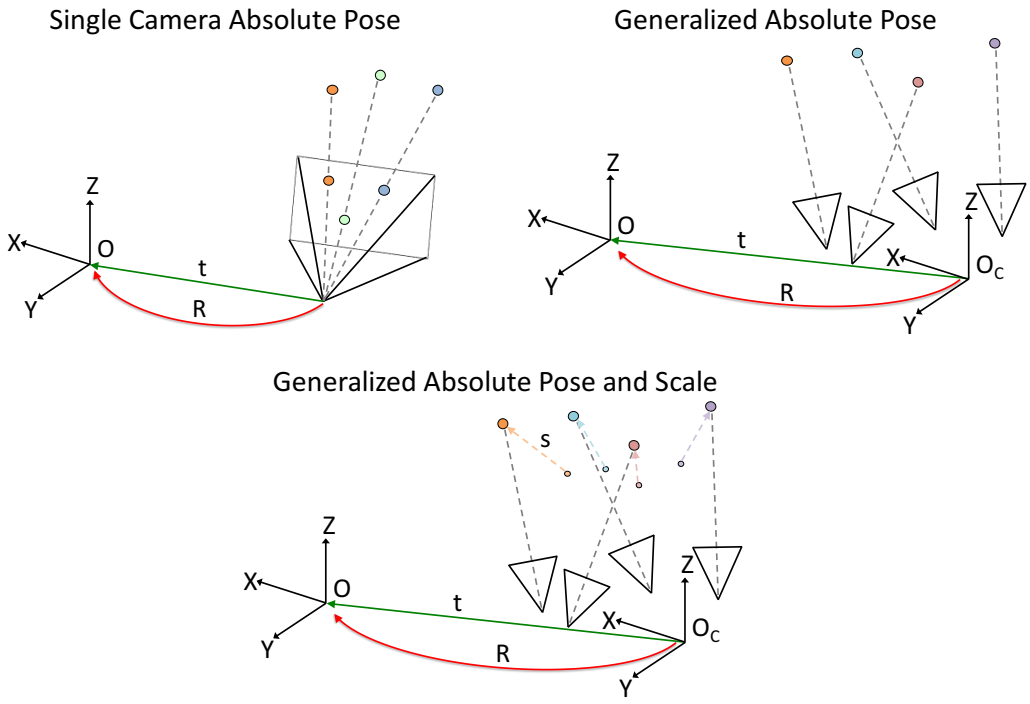

Figure 4.1: The absolute camera pose problem determines a camera's position and orientation with respect to a coordinate system with an origin O from correspondences between 2D image points and 3D points. In this chapter, we present a formulation of the absolute pose problem that recovers the pose for single-cameras (top-left), pose of multi-camera rigs (top-right), and pose and scale change for multi-camera rigs (bottom).

that minimize the reprojection error:

$$C(R,t) = \sum_{i=1}^{3} ||\overline{x}_i - \frac{1}{\alpha_i}(RX_i + t)||^2 \tag{4.2}$$

In the case of a calibrated camera, several minimal methods exist to efficiently solve the P3P problem [25, 34, 45]. The efficiency of these methods makes P3P vital to applications such as SfM and visual odometry where noise and outliers exist and P3P may be used within a RANSAC scheme to robustly and efficiently determine the camera pose.

4.1 Generalization to 7 d.o.f.

When information from multiple images is available, PnP methods are no longer suitable and few methods exist that are able to jointly utilize information from many cameras simultaneously. As illustrated in Figure 4.1 (center), multiple cameras (or multiple images from a single moving camera) can be described with the generalized camera model [63]. The generalized camera model represents a set of observations by the viewing ray origins and directions. For multi-camera systems, these values may be determined from the positions and orientations of cameras within the rig. The generalized camera model considers the viewing rays as static with respect to a common coordinate system (*cf*. O_C in Figure 4.1 center, right). Using the generalized camera model, we may extend the reprojection constraint of Eq. (4.1) to multiple cameras:

$$c_i + \alpha_i \overline{x}_i = RX_i + t, \qquad i = 1, \ldots, n \tag{4.3}$$

where c_i is the origin of the feature ray $\overline{x}_i$ within the generalized camera model. This representation assumes that the scale of the generalized camera is equal to the scale of the 3D points (*e.g.*, that both have metric scale). In general, the internal scale of each generalized camera is not guaranteed to be consistent with the scale of the 3D points. Consider a multi-camera system on a car that we want to localize to a point cloud created from Google Street View images. While we may know the metric scale of the car's multi-camera rig, it is unlikely we have accurate scale calibration for the Google Street View point cloud, and so we must recover the scale transformation between the rig and the point cloud in addition to the rotation and translation. This leads to the following reprojection constraint:

$$sc_i + \alpha_i \overline{x}_i = RX_i + t, \qquad i = 1, \ldots, n \tag{4.4}$$

where α_i is a scalar which stretches the image ray such that it meets the world point X_i such that $\alpha_i = ||RX_i + t - sc_i||$. It is trivial to see that the generalized pose problem occurs when $s = 1$ and the single-camera absolute pose problem occurs when $c_i = \mathbf{0} \ \forall i$. In this way, it is a complete generalization of the absolute pose problem.

By extending Eq (4.1) to multi-camera systems and scale transformations, we have generalized the PnP problem to the generalized pose-and-scale (gPnP+s) problem in Eq (4.4) shown in Figure 4.1 (right). The goal of the gPnP+s problem is to determine the pose and internal scale of a generalized camera with respect to n known 3D points. This is equivalent to aligning the two coordinate systems that define the generalized camera and the 3D points, and thus the solution to the gPnP+s problem is a 7 d.o.f. similarity transformation.

The generalized pose-and-scale problem is one that frequently arises in SLAM and SfM. It is impossible to recover scale from images alone. Scale may be recovered in special cases, for instance, when observing an object with known metric dimensions or with the aid of sensor measurements, but these cases are not common. As a result, the relative scale must be reconciled, for example, during loop closure for SLAM or when registering multiple reconstructions from SfM. The generalized pose-and-scale problem aims to compute a similarity transformation that will align two coordinate systems such that it minimizes reprojection error from 2D-3D correspondences. By minimizing reprojection error, gPnP+s is well-suited for multi-view geometry tasks such as model-merging.

The solution proposed in this chapter solves the generalized pose-and-scale problem, estimating rotation, translation, and scale directly given n 2D-3D observations. The solution is general in that the technique may be used to solve the single camera absolute pose problem, the generalized (multi-camera) absolute pose problem, or the generalized pose-and-scale problem without modification. Our approach is $O(n)$ in the number of observations, making it useful for real-time applications, and does not require initialization.

Additionally, we solve for all minima of our least squares cost function simultaneously instead of a single local minimum. Experiments on synthetic and real data show that our method is more accurate and scalable than other alignment methods.

4.2 An L_2 Optimal Solution

We use the unit-norm quaternion parameterization of the rotation matrix such that R can be formed with four unknowns representing the quaternion $q = [q_0,\ q_1,\ q_2,\ q_3]^\top$. The rotation matrix R may be written as:

$$R = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2q_1q_2 - 2q_0q_3 & 2q_1q_3 + 2q_0q_2 \\ 2q_1q_2 + 2q_0q_3 & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2q_2q_3 - 2q_0q_1 \\ 2q_1q_3 - 2q_0q_2 & 2q_2q_3 + 2q_0q_1 & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}. \tag{4.5}$$

Unlike the Cayley-Gibbs-Rodriguez rotation parameterization used in [79], the unit-norm quaternion does not have any singularities. As shown in Section 4.3, this leads to more accurate results. When considering all n correspondences, there exists $8+n$ unknown variables (4 for rotation, 3 for translation, 1 for scale, and 1 unknown depth per observation). The gPnP+s problem can be formulated from Eq. (4.4) as a non-linear least-squares minimization such that the sum of squared measurement errors is minimized. Thus, we aim to minimize the cost function:

$$C(R, t, s) = \sum_{i=1}^{n} ||\overline{x}_i - \frac{1}{\alpha_i}(RX_i + t - sc_i)||^2. \tag{4.6}$$

This non-linear least squares problem can be solved with iterative methods such as Gauss-Newton; however, these techniques are sensitive to initialization and only converge to a single local minimum. In this section, we describe our method for directly solving

for all minima of a slightly modified cost function without the need for initialization.

The geometric constraint equation of Eq. (4.4) leads to a non-linear system of equations that can be minimized by a least squares solver that minimizes Eq. (4.6). We would instead like to rewrite this system of equations in terms of fewer unknowns. Specifically, we can rewrite this equation solely in terms of the unknown rotation, R. When we relax the constraint that $\alpha_i = ||RX_i + t - sc_i||$ and treat each α_i as a free variable, α_i, s, and t appear linearly and can be easily reduced from Eq. (4.6). Note that this relaxation is reasonable since solving the optimality conditions results in $\alpha_i^* = z_i^\top (RX_i + t - sc_i)$ where z_i is $\overline{x_i}$ corrupted by measurement noise.

We begin by rewriting our system of equations from Eq. (4.4) in matrix-vector form:

$$\underbrace{\begin{bmatrix} \overline{x_1} & & & c_1 & -I \\ & \ddots & & \vdots & \vdots \\ & & \overline{x_n} & c_n & -I \end{bmatrix}}_{\text{A}} \underbrace{\begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \\ s \\ t \end{bmatrix}}_{\text{x}} = \underbrace{\begin{bmatrix} R & & \\ & \ddots & \\ & & R \end{bmatrix}}_{\text{W}} \underbrace{\begin{bmatrix} X_1 \\ \vdots \\ X_n \end{bmatrix}}_{\text{b}} \tag{4.7}$$

$$\Leftrightarrow Ax = Wb, \tag{4.8}$$

where A and b consist of known and observed values, x is the vector of unknown variables we will eliminate from the system of equations, and W is the block-diagonal matrix of the unknown rotation matrix. From Eq. (4.7), we can create a simple expression for x:

$$x = (A^\top A)^{-1} A^\top W b = \begin{bmatrix} U \\ S \\ V \end{bmatrix} Wb. \tag{4.9}$$

We have partitioned $(A^\top A)^{-1}A^\top$ into constant matrices U, S, and V such that the depth, scale, and translation parameters are functions of U, S, and V respectively. Matrices U, S, and V can be efficiently computed in closed form by exploiting the sparse structure of the block matrices (see Appendix A for the full derivation). Note that α_i, s, and t may now be written concisely as linear functions of the rotation:

$$\alpha_i = u_i^\top Wb \tag{4.10}$$

$$s = SWb \tag{4.11}$$

$$t = VWb, \tag{4.12}$$

where $u_i^\top$ is the i-th row of U. Through substitution, the geometric constraint equation (4.4) can be rewritten as:

$$\underbrace{SWb}_{s} c_i + \underbrace{u_i^\top Wb}_{\alpha_i} \overline{x_i} = RX_i + \underbrace{VWb}_{t}. \tag{4.13}$$

This new constraint is quadratic in the four unknown rotation variables given by the unit-norm quaternion representation.

4.2.1 A New Least Squares Cost Function

The new geometric constraint equation (4.13) assumes noise-free observations. We assume that each observation is noisy, with zero mean noise η_i. We can denote our noisy observations as $\overline{z_i} = \overline{x_i} + \eta_i$. We can rewrite our measurement constraint in terms of our

noisy observation:

$$SWbc_i + u_i^\top Wb(\overline{z_i} - \eta_i) = RX_i + VWb \tag{4.14}$$

$$\Rightarrow \eta_i' = SWbc_i + u_i^\top Wb\overline{z_i} - RX_i - VWB, \tag{4.15}$$

where η_i' is a zero-mean noise term that is a function of η_i (but whose covariance depends on the system parameters, as noted by Hesch and Roumeliotis [34]). We evaluate u_i, S, and V at $\overline{x_i} = \overline{z_i}$ without loss of generality. Observe that u_i can be eliminated from Eq. 4.15 by noting that:

$$UWb = \begin{bmatrix} \overline{z_i}^\top & & \\ & \ddots & \\ & & \overline{z_n}^\top \end{bmatrix} Wb - \begin{bmatrix} \overline{z_1}^\top c_1 \\ \vdots \\ \overline{z_n}^\top c_n \end{bmatrix} SWb + \begin{bmatrix} \overline{z_1}^\top \\ \vdots \\ \overline{z_n}^\top \end{bmatrix} VWb \tag{4.16}$$

$$\Rightarrow u_i^\top Wb = \overline{z_i}^\top RX_i - \overline{z_i}^\top c_i SWbc_i + \overline{z_i}^\top VWb. \tag{4.17}$$

Through substitution, Eq. (4.15) can be refactored such that:

$$\eta_i' = (\overline{z_i z_i}^\top - I_3)(RX_i - SWbc_i + VWb). \tag{4.18}$$

Eq. (4.18) allows the gPnP+s problem to be formulated as an unconstrained least-squares minimization in 4 unknown rotation parameters. We formulate our new least squares cost function, C', as the sum of the squared constraint errors from Eq. (4.18):

$$C'(R) = \sum_{i=1}^{n} ||(\overline{z_i z_i}^\top - I_3)(RX_i - SWbc_i + VWb)||^2 \tag{4.19}$$

$$= \sum_{i=1}^{n} \eta_i'^\top \eta_i'. \tag{4.20}$$

Thus, we have reduced the number of unknowns in our system from $8 + n$ to 4. This is an important part of our formulation, as it allows the size of the system we solve to be independent of the number of observations and thus scalable.

4.2.2 Macaulay Matrix Solution

We have reduced our original geometric constraint of Eq. (4.4) to a least-squares minimization as a function of only the four unknown rotation parameters in Eq. (4.19). That is, we wish to find the unknown rotation parameters q_0, q_1, q_2, q_3 such that C' is minimized. This polynomial is of a similar form to the DLS PnP algorithm [34], but uses a singularity-free rotation parameterization. Thus, we can solve our least squares system of Eq. (4.19) with the same technique on our modified equations.

We employ the Macaulay matrix [56] to determine the solution to our polynomial system. This matrix is formed from the partial derivatives of our cost function C' with respect to the four unknown rotation parameters. These equations each equal zero when C' is minimal, so the roots of these polynomials produce the solution to our system. We consider one additional equation to constrain the quaternion to have a unit norm so it is a valid rotation:

$$q^\top q = 1 \tag{4.21}$$

The Macaulay matrix, M, formed from the four partial derivatives and the additional linear equation, forms an extended polynomial system as a 200×200 matrix that contains coefficients to 200 monomials of the polynomial system.

Using the Schur complement trick, the Macaulay resultant matrix can be reduced to a 40×40 matrix whose eigenvectors correspond to the monomials of our cost function. The unknown rotation variables appear in these monomials, and can be directly extracted from the eigenvectors. This leads to 40 real and imaginary critical points,

though the number of solutions can be reduced by considering only real solutions that place points in front of cameras. In practice, when using $n \geq 6$ points there exists only one valid minimum. After obtaining all minima, we evaluate the cost function Eq. (4.19) to determine the best orientation and compute the corresponding scale and translation through back substitution.

4.2.3 Gröbner Basis Solution

An alternative method for solving polynomial systems is the Gröbner basis technique [47]. The Gröbner basis method operates by solving the polynomial system in a finite prime field $\mathbb{Z}_p$ where exact zero calculations may take place. Typically $\mathbb{Z}_p$ is the field of integers modulo a very large prime number [44]. A so-called "elimination template" is constructed such that LU decomposition (or Gauss-Jordan) reveals an action matrix whose set of eigenvectors contains the solution set. Often the most expensive part of this operation is the eigen-decomposition, so reducing the number of candidate solutions is vital to designing an efficient Gröbner basis solver[1].

We created a Gröbner basis solver with an automatic generator similar to [47] while taking advantage of several additional forms of improvement. Following the solver of Kneip *et al.* [44], we reduce the size of the Gröbner basis by only choosing random values in $\mathbb{Z}_p$ that correspond to valid configurations for the generalized pose-and-scale problem. Next, we eliminate double-roots by exploiting the 2-fold symmetry technique used in [7, 44, 92]. This technique requires that all polynomials contain only even or only odd-degree monomials. The first order optimality constraints (formed from the partial derivitives of C'; see Section 4.2.2) contain only uneven monomials; however, the unit-norm quaternion constraint of Eq. (4.21) contains even monomials. By modifying the

[1]See [47] for more details about Gröbner basis techniques.

unit-norm quaternion constraint of Eq. (4.21) to the squared norm:

$$(q^\top q - 1)^2 = 0 \tag{4.22}$$

we obtain equations with only odd monomials when considering the first order derivitives of the squared unit-norm constraint. Our final polynomial system is then:

$$\frac{\partial C'}{\partial q_i} = 0 \qquad i = 0, 1, 2, 3 \tag{4.23}$$

$$(q^\top q - 1) q_i = 0 \qquad i = 0, 1, 2, 3. \tag{4.24}$$

These eight third-order polynomials contain only uneven degree monomials, and so we can apply the 2-fold symmetry technique proposed by Ask *et al.* [7]. As with the UPnP method [44], applying these techniques to our Gröbner basis solver creates a 141×149 elimination template with an action matrix that is 8×8. Both the elimination template and the action matrix are dramatically smaller than with the Macaulay Matrix solution, resulting in a significant speedup (see Section 4.3.4).

It is interesting to note that this formulation implicitly solves an equivalent Lagrangian formulation:

$$L(q,\ \lambda) = C' + \lambda(q^\top q - 1)^2 \tag{4.25}$$

$$q,\ \lambda = \arg\min_{q}\ \arg\max_{\lambda}\ L \tag{4.26}$$

When formulating the problem this way, we additionally need to compute the gradient of L with respect to λ:

$$\frac{\partial L}{\partial \lambda} = (q^\top q - 1)^2 = 0. \tag{4.27}$$

Eq. (4.24) implies that either $(q^\top q - 1)^2 = 0$ or all $q_i = 0$. Since all quaternion parameters

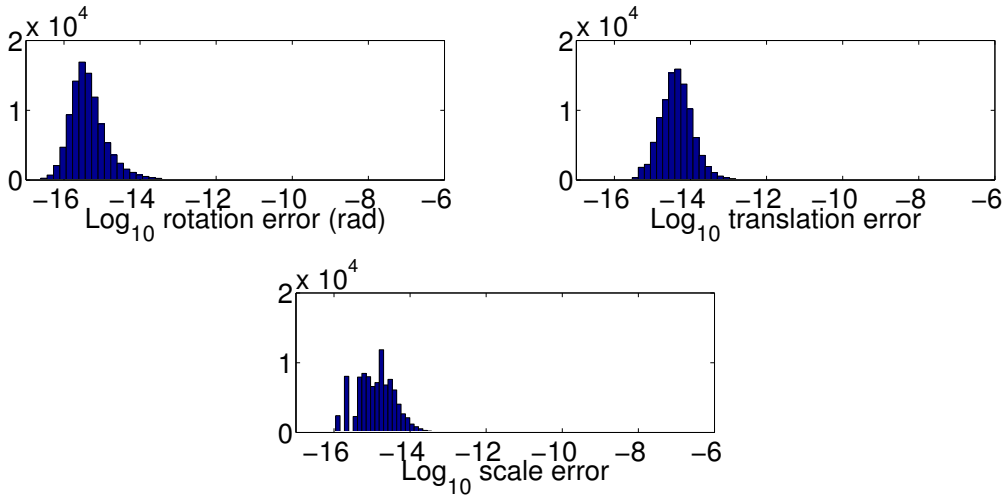

Figure 4.2: Histograms of numerical errors in the computed similarity transforms based on 10^5 random trials with the minimal 4 correspondences. Our algorithm demonstrates excellent numeric stability.

may never simultaneously be zero, $(q^\top q - 1)^2$ must be equal to zero and thus $\dfrac{\partial L}{\partial \lambda} = 0$, implying that the Lagrange multiplier does not matter. This is because Eq. (4.24) implies that Eq. (4.27) will be solved.

4.3 Experimental Evaluation

4.3.1 Numerical Stability

We tested the numerical stability of our solution over 10^5 random trials. We generated uniformly random camera configurations that placed cameras (*i.e.*, ray origins) in the cube $[-1, 1] \times [-1, 1] \times [-1, 1]$ around the origin. 3D points were randomly placed in the volume $[-1, 1] \times [-1, 1] \times [2, 4]$. Ray directions were computed as unit vectors from camera origins to 3D points. An identity similarity transformation was used (*i.e.*, $R = I$, $t = 0$, $s = 1$). For each trial, we computed solutions using the minimal 4 correspondences. We calculated the angular rotation error, the translation error, and the scale error for

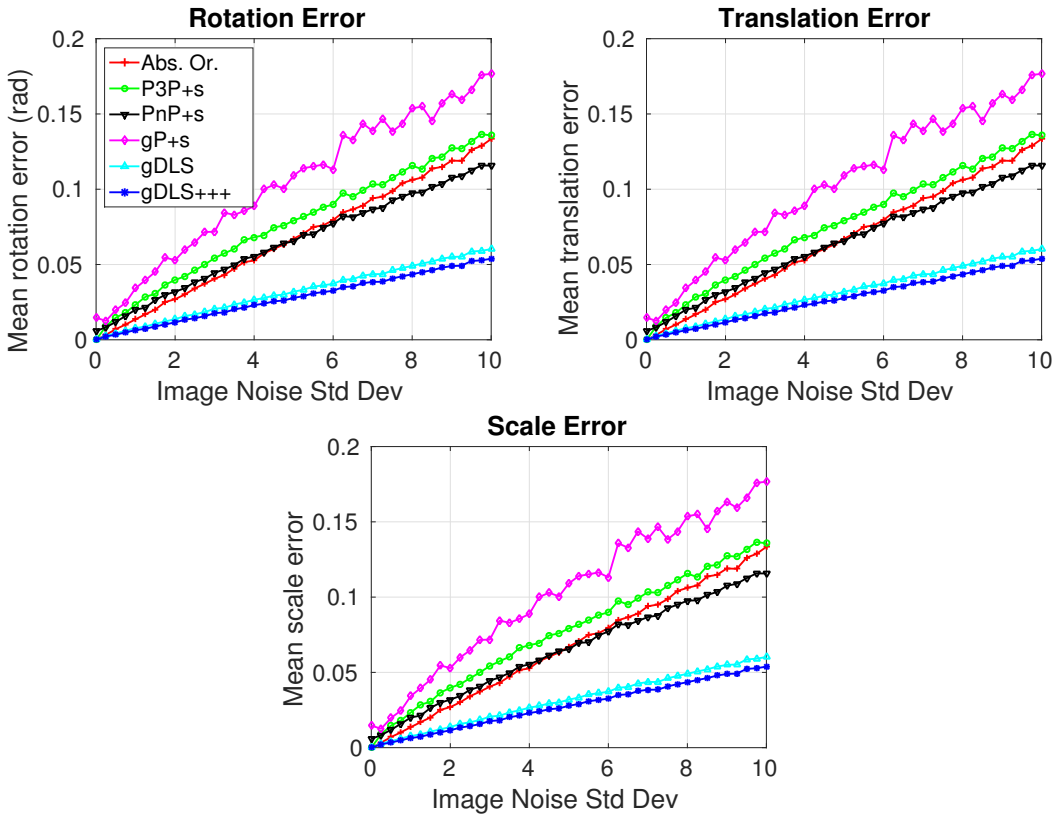

Figure 4.3: We compared similarity transform algorithms with increasing levels of image noise to measure the pose error performance: the absolute orientation algorithm of Umeyama [86], P3P+s, PnP+s, gP+s[87], and our algorithm, gDLS. Each algorithm was run with the same camera and point configuration for 1000 trials per noise level. Our algorithm has mean better rotation, translation, and scale errors for all levels of image noise.

each trial, and plot the results in Figure 4.2. The errors are very stable, with 98% of all errors less than 10^{-12}.

4.3.2 Simulations With Noisy Synthetic Data

We performed two experiments with synthetic data to analyze the performance of our algorithm as the amount of image noise increases and as the number of correspondences increases. For both experiments we use a focal length of 800 and $[640, 480]$ resolution. Two cameras are placed randomly in the cube $[-1, 1] \times [-1, 1] \times [-1, 1]$ around the origin

with three 3D points randomly placed in the volume $[-1,1]\times[-1,1]\times[2,4]$. Both cameras observe each 3D point, so there are six total 2D-3D observations. Using the known 2D-3D correspondences, we apply a similarity transformation with a random rotation in the range of $[-30,30]$ degrees about each of the x, y, and z axes, a random translation with a distance between 0.5 and 10, and a random scale change between 0.1 and 10. We measure the performance of the following similarity transform algorithms:

- **Absolute Orientation:** The absolute orientation method of Umeyama [86] is used to align the known 3D points to 3D points triangulated from 2D correspondences. This algorithm is only an alignment method and does not utilize any 2D correspondences.

- **P3P+s:** The P3P algorithm of Kneip *et al.* [45] is used to localize the first camera and the corresponding rotation and translation is used for the similarity transformation. The scale is then estimated from the median estimate from triangulated point matches. This process is repeated for all cameras, and the camera localization and scale estimation that yields the largest number of inliers is used as the similarity transformation.

- **PnP+s:** The similarity transformation is computed the same way as P3P+s, but the DLS PnP algorithm of Hesch and Roumeliotis [34] is used to localize each camera instead of P3P[2]. PnP+s uses $n \geq 3$ 2D-3D correspondences, whereas P3P+s can only use 3.

- **gP+s:** The minimal solver of Ventura *et al.* [87] is used with 2D-3D correspondences from all cameras. While the algorithm is intended for the minimal case

[2]We found that DLS [34] performed comparably to alternatively algorithms such as OPnP [92] in the context of PnP+s.

of $n = 4$ correspondences, it can compute an overdetermined solution for $n \geq 4$ correspondences.

- **gDLS:** The algorithm presented in [79], which uses $n \geq 4$ 2D-3D correspondences from all cameras.

- **gDLS+++:** The algorithm presented in this chapter, which is an extension of the gDLS algorithm [79]. This method uses $n \geq 4$ 2D-3D correspondences from all cameras.

After running each algorithm on the same camera and point configuration, we calculate the rotation, translation, and scale errors with respect to the known similarity transformation.

Image noise experiment: For our first experiment, we evaluated the similarity transformation algorithms under increased levels of image noise. Using the configuration described above, we increased the image noise from 0 to 10 pixels standard deviation, and ran 1000 trials at each level. Our algorithm outperforms each of the other similarity transformation algorithms for all levels of image noise, as shown in Figure 4.3. The fact that our algorithm returns all minima of our modified cost function is advantageous under high levels of noise, as we are not susceptible to getting stuck in a bad local minimum. This allows our algorithm to be very robust to image noise as compared to other algorithms.

Scalability experiment: For the second experiment, we evaluate the rotation, translation, and scale error as the number of 2D-3D correspondences increases. We use the same camera configuration described above, but vary the number of 3D points used to compute the similarity transformation from 4 to 1000. Each 3D point is observed by both cameras. We ran 1000 trials for each number of correspondences used with a Gaussian noise level of 0.5 pixels standard deviation for all trials. We did not use the

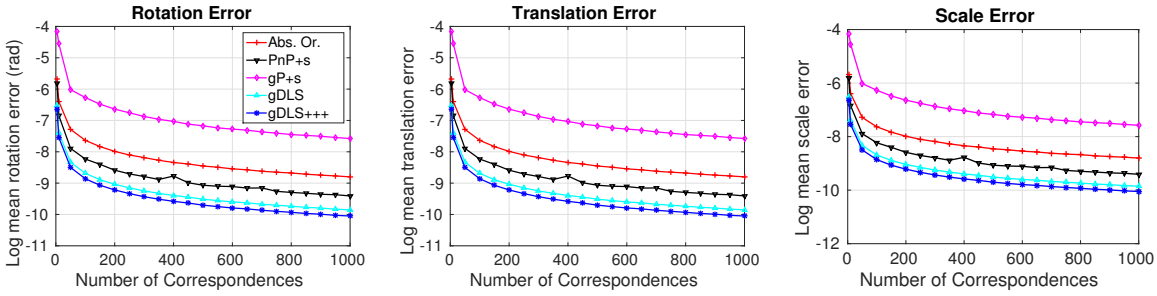

Figure 4.4: We measured the accuracy of similarity transformation estimations as the number of correspondences increased. The mean of the log rotation, translation, and scale errors are plotted from 1000 trials at each level of correspondences used. A Gaussian image noise of 0.5 pixels was used for all trials. We did not use P3P+s in this experiment because P3P only uses 3 correspondences. Our algorithm has better accuracy for all number of correspondences used and a runtime complexity of $O(n)$, making it ideal for use at scale.

P3P+s algorithm for this experiment since P3P is a minimal solver and cannot utilize the additional correspondences. Although gP+s is a minimal solver, it can utilize all n correspondences in an overdetermined solution. The accuracy of each similarity transformation algorithm as the number of correspondences increases is shown in Figure 4.4. Our algorithm performs very well as the number of correspondences increases, and is more accurate than alternative algorithms for all numbers of correspondences tested. Further, our algorithm is $O(n)$ so the performance cost of using additional correspondences is favorable compared to the alternative algorithms (see Section 4.3.4 for a full runtime analysis).

4.3.3 SLAM Registration With Real Images

We tested our new solver for registration of a SLAM reconstruction with respect to an existing SfM reconstruction using the indoor dataset from [87]. This dataset consists of an indoor reconstruction with precise 3D and camera position data obtained with an ART-2 optical tracker. Several image sequences in this environment were run

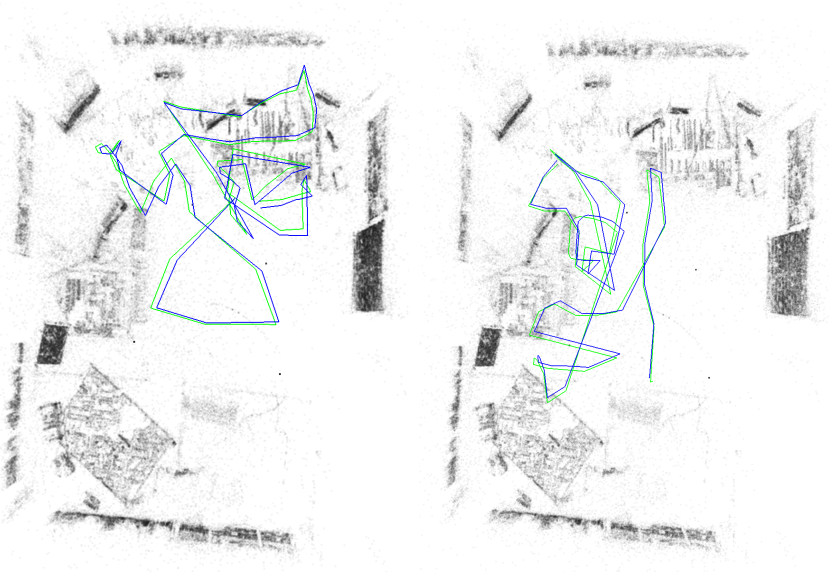

Figure 4.5: In our real data experiments we compute the similarity transformation that aligns cameras from a SLAM system (blue) to a preexisting SfM reconstruction using 2D-3D correspondences. The ground truth positions (green) were recorded with a high-accuracy ART-2 tracker.

through a real-time keyframe-based SLAM system to obtain a local tracking sequence that can be registered to the ground-truth environment via a similarity transform (see Figure 4.5). SIFT keypoints were used to establish 2D-3D correspondences using approximate nearest-neighbor techniques and a ratio test. We compare our method to several other techniques for registering SLAM maps to a global point cloud: the absolute orientation algorithm [86], P3P+s, PnP+s (using 4 correspondences), gP+s [87], and gDLS [79] as described in Section 4.3.2. All algorithms are used in a PROSAC loop except for the absolute orientation algorithm which is used in a RANSAC loop. The absolute orientation algorithm does not use feature matches (it only aligns 3D point clouds) and thus cannot utilize matching scores in a PROSAC loop. We compare these algorithms to our gDLS+++ algorithm when no refinement is performed after RANSAC/PROSAC for any of the algorithms.

We compute the average position error of all keyframes with respect to the ground

Table 4.1: Average position error in centimeters for aligning a SLAM sequence to a pre-existing SfM reconstruction. An ART-2 tracker was used to provide highly accurate ground truth measurements for error analysis. Camera positions were computed using the respective similarity transformations and the mean camera position error of each sequence is listed below. Both the minimal version of our solver, gDLS4, and the nonminimal gDLS10 (both shown in bold below) outperform the alternative methods.

Sequence	# Images	Abs. Ori. [86]	P3P+s	PnP+s	gP+s[87]	gDLS [79]	gDLS+++	After BA
office1	9	6.37	6.14	4.38	6.12	3.97	3.68	3.61
office2	9	8.09	7.81	6.90	9.32	5.89	5.59	5.57
office3	33	8.29	9.31	8.89	6.78	6.08	4.91	4.86
office4	9	4.76	4.48	3.98	4.00	3.81	3.09	3.04
office5	15	3.63	3.42	3.39	4.75	3.39	3.17	3.14
office6	24	5.15	5.23	5.01	5.91	4.51	4.35	4.31
office7	9	6.33	7.08	7.16	7.07	4.65	2.99	2.72
office8	11	4.72	4.85	3.62	4.59	2.85	2.30	2.12
office9	7	8.41	8.44	4.08	6.65	3.19	2.25	2.25
office10	23	5.88	6.60	5.73	5.88	4.94	4.68	4.61
office11	58	5.19	4.85	4.80	6.74	4.77	4.66	4.57
office12	67	5.53	5.20	4.97	4.86	4.81	4.45	4.44

truth data. The position errors, reported in centimeters, are shown in Table 4.1. Our gDLS+++ solver give higher accuracy results for every image sequence tested compared to alternative algorithms. By using the generalized camera model, we are able to exploit 2D-3D constraints from multiple cameras at the same time as opposed to considering only one camera (such as P3P+s and PnP+s). This allows the similarity transformation to be optimized for all cameras and observations simultaneously, leading to high-accuracy results.

We additionally show the results of gDLS+++ with bundle adjustment applied after estimation (labeled "After BA" in Table 4.1). In all datasets, our results are very close to the optimal results after bundle adjustment, and typically bundle adjustment converges after only one or two iterations. This indicates that the gDLS+++ algorithm is very close to the geometrically optimal solution in terms of reprojection error. Further, our singularity-free rotation parameterization prevents numerical instabilities that arise as the computed rotation approaches a singularity, leading to more accurate results than

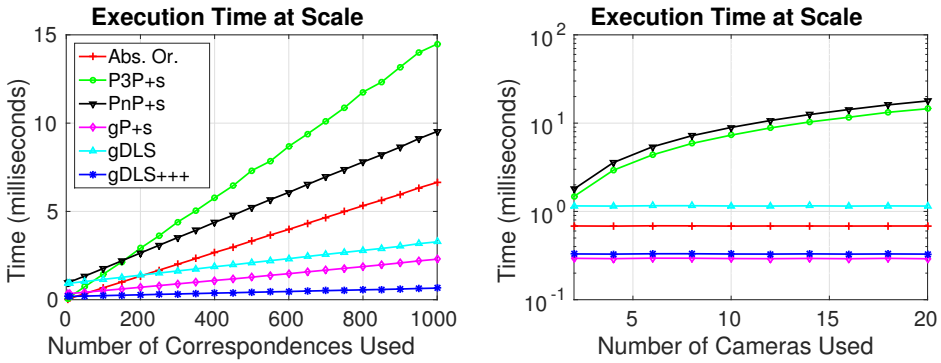

Figure 4.6: **Top:** We plot the mean execution time while increasing the number of 2D-3D correspondences used and keeping the number of cameras constant at 2. Our gDLS+++ method is slightly slower than gP+s [87] in the minimal case, though our method scales favorably and is much more accurate at scale. **Bottom:** The number of cameras was increased while using the 100 2D-3D correspondences (each point is seen in every camera). The runtimes of the absolute orientation method [86], gP+s [87], gDLS [79] and our gDLS+++ method are independent of the number of cameras.

the gDLS [79] algorithm.

4.3.4 Runtime Analysis

In this section we present a runtime analysis of each similarity transformation algorithm when registering n points and m cameras. The absolute orientation algorithm requires aligning sets of 3D points, making it $O(n)$. In theory, the covariance matrix used to compute the least-squares solution has a lower bound of $O(n)$; however, in practice it is often slower. The P3P+s algorithm relies on the extremely efficient P3P algorithm (which can be considered to run in constant time); however, in order to recover scale it must triangulate points across all cameras, leading to $O(n)$ complexity. Further, P3P+s computes the similarity transformation by localizing each camera and estimating scale. Thus, for m cameras the expected runtime is $O(mn)$. The PnP+s algorithm operates the same way as P3P, though the PnP algorithm is at best $O(n)$, resulting in an overall runtime of $O(mn^2)$. In practice, the PnP+s algorithm outperforms the P3P+s algorithm as the number of points increases. This is because the P3P algorithm returns four solu-

Table 4.2: Results of our Hierarchical SfM pipeline on several large scale datasets. Our method is extremely efficient and is able to reconstruct more cameras than the Divide-and-Conquer method of Bhowmick *et al.* [9].

Dataset	N_{cam}	Incremental SfM [89]		DISCO [20]		Hierarchical SfM [9]		Ours		
		N_{cam}	Time (min)	N_{cam}	Time (min)	N_{cam}	Time (min)	N_{cam}	BA Its	Time (min)
Colosseum	1164	1157	9.85	N/A	N/A	1032	3	1097	1	2.6
St Peter's Basilica	1275	1267	9.71	N/A	N/A	1236	4	1256	1	3.7
Dubrovnik	6845	6781	16.8	6532	275	N/A	N/A	6677	2	8.9
Rome	15242	15065	100.17	14754	792	10534	27	12329	2	22

tions, while the DLS PnP algorithm returns only one solution in most cases. All n points must be triangulated for each solution, leading to an increased runtime for PnP+s. The gP+s algorithm requires computing the null space of a matrix that is of size $2n$, which is $O(n)$ in theory though efficient in practice even for large n. The gP+s algorithm is only slightly more efficient than the gDLS+++ algorithm; however, as shown in Sections 4.3.2 and 4.3.3 it is less accurate than our algorithm.

As described in Section 4.2.1, our algorithm is $O(n)$ in the number of points, making the runtime favorable as the number of points increases (*cf.* Figure 4.6). Additionally, our algorithm's runtime is independent of the number of cameras. In our experiments conducted on a 3.6 GHz Intel i7, we observed a mean runtime of roughly 151 μs over 10^5 trials when using four points. When using 10 points, the mean runtime increased to roughly 166 μs. This is nearly six times faster than the original gDLS algorithm [79] which has a similar runtime to the Macaulay Matrix solution proposed in Section 4.2.2. This efficiency allows our method to be used in real-time within a RANSAC loop. Additionally, the fact that our method can be used with only four correspondences allows for the theoretical convergence rate of RANSAC to remain low compared to algorithms that require more correspondences.

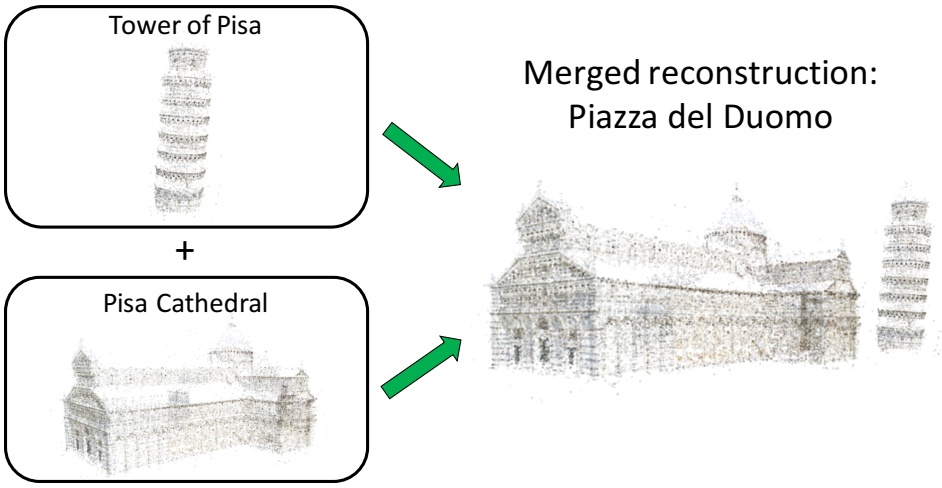

Figure 4.7: We present a novel hierarchical SfM pipeline that merges reconstructions accurately and efficiently. Our model-merging method computes a similarity transformation that minimizes reprojection error and is used to align SfM reconstructions that have scene overlap.

4.4 A Hierachical SfM Pipeline

We demonstrate the viability of our method for SfM model-merging in a novel hierarchical SfM pipeline. In contrast to incremental or global SfM methods, hierarchical SfM operates by first partitioning the input data into subsets that can be individually reconstructed (*cf*. Figure 4.7). After these subsets are reconstructed, they are merged together in a tree-like fashion as shown in Figure 4.8. The subset reconstructions can be built and merged in parallel making hierarchical SfM typically much faster than incremental SfM.

Our hierarchical SfM pipeline draws inspiration from the Divide-and-Conquer method of Bhowmick *et al*. [9]. We begin by partitioning the input dataset into subsets using normalized graph cuts [70]. Each of the subsets is then individually reconstructed in parallel with VisualSfM [89]. We merge subset reconstructions using the gDLS+++ algorithm to align the reconstructions with a similarity transformation. No bundle adjustment is performed after subsets are merged. This merging process is repeated until a single re-

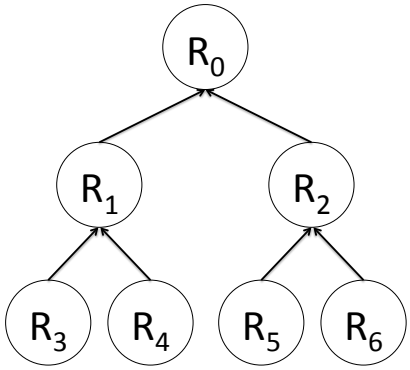

Figure 4.8: Hierarchical SfM splits the input set of cameras into subsets that are individually reconstructed and successively merged in a tree-like procedure. Our hierarchical SfM pipeline uses the gDLS+++ algorithm for model-merging.

construction remains (or no more reconstructions can be successfully merged). As a final step, we run bundle adjustment on the entire reconstruction.

Another way to view this hierarchical method is to consider each of the partitioned subsets as distributed cameras. Each distributed camera is recovered individually using VisualSfM [89], then the we localize one distributed camera to the 3D points of another using gDLS+++ to solve the generalized absolute pose-and-scale problem. This is analogous to localizing a single camera to a reconstruction in incremental SfM by using P3P to solve the absolute pose problem. Our hierarchical SfM pipeline repeats this process, localizing all of the other distributed cameras. In this way, we are able to use the distributed camera as a building block for this SfM pipeline. As a final step, we perform a single bundle adjustment on the entire reconstruction.

We compare our hierarchical SfM pipeline to Incremental SfM (VisualSfM [89]), the DISCO pipeline of Crandall *et al*. [20], and the hierarchical SfM pipeline of Bhowmick *et al*. [9] run on several large-scale SfM datasets and show the results in Table 4.2. It is clear both DISCO and incremental SfM are slower than hierarchical SfM, though they are often able to reconstruct more cameras. The efficiency of hierarchical SfM is due of the parallel nature of the reconstruction and merging procedure.

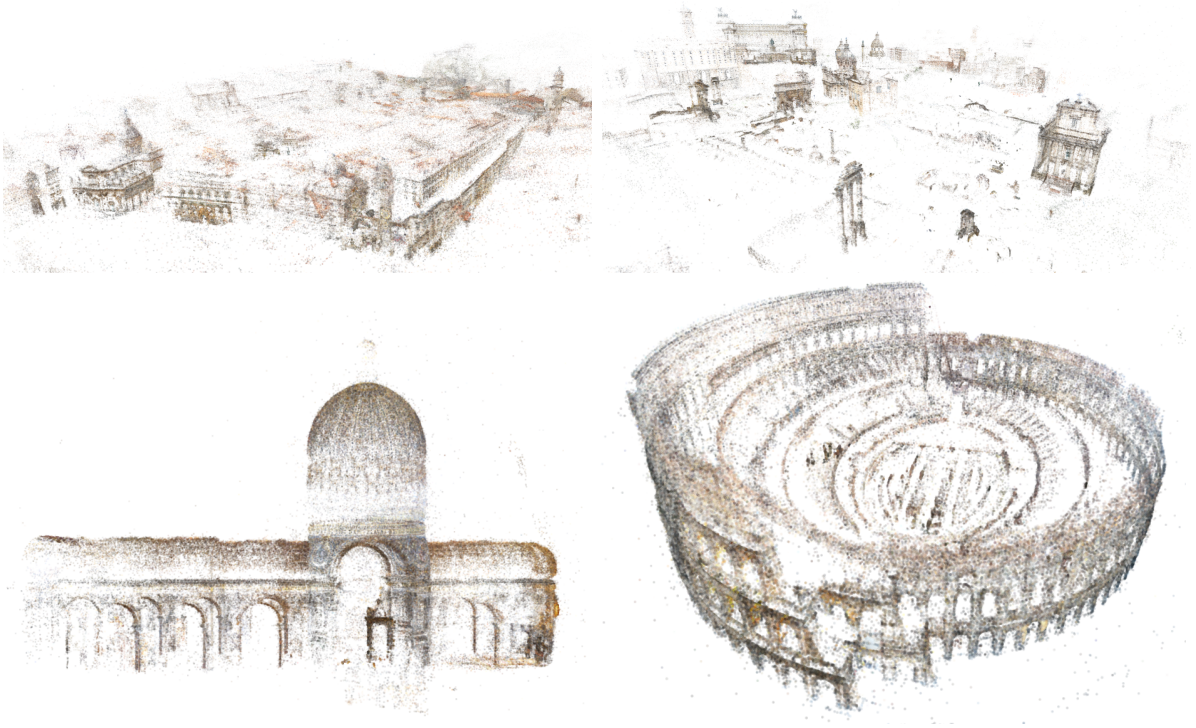

Figure 4.9: Final reconstructions of the Dubrovnik (top-left), Central Rome (top-right), St Peters (bottom-left), and Colosseum (bottom-right) datasets computed with our hierarchical SfM pipeline. Our pipeline produces high quality visual results at state-of-the-art efficiency (*cf*. Table 4.8).

When compared to the hierarchical SfM pipeline of Bhowmick *et al*. [9], our method is typically 7-20% faster. Both methods use the same normalized graph cut procedure to determine image subsets and use Visual SfM [89] to reconstruct the subsets. Thus, the only difference between the methods is in the technique used for model-merging. By using gDLS+++ for model-merging, our method produces high quality models more efficiently than the hierarchical pipeline of Bhowmick *et al*. [9]. As shown in Table 4.8, the final bundle adjustment for our pipeline requires no more than 2 iterations, indicating that the gDLS+++ method is able to merge reconstructions extremely accurately. We show the high quality visual results of our hierarchical SfM pipeline in Figure 4.9.

4.5 Discussion

In this chapter, we have provided a full generalization of the absolute camera pose problem and provided an efficient and accurate solution method. This generalization extends the standard reprojection of Eq. (4.1) to handle multiple cameras and scale changes simultaneously (Eq. (4.4)) and thus is suitable for use with the distributed camera. We provide a technique for solving the generalized pose-and-scale problem such that 7 d.o.f. similarity transformations that minimize reprojection error may be efficiently computed. Our method, gDLS+++, is general, accurate, and efficient. It can handle the minimal case of $n = 4$, as well as the overdetermined case of $n > 4$. We formulate a least squares cost function that can be solved efficiently as a system of third degree polynomials, resulting in a system that is $O(n)$ in the number of correspondences. This makes it applicable for real-time frameworks and is useful, for example, for loop closure with SLAM and visual odometry. We have evaluated our method on synthetic data to show the numerical stability, accuracy under image noise, and scalability of our method. We validated our method with experiments using real data which shows that our method is more accurate than other methods when computing the similarity transformation for registering reconstructions. Finally, we presented a hierarchical SfM pipeline that utilizes gDLS+++ to solve large-scale SfM problems extremely efficiently. This pipeline is extremely scalable, and is able to reconstruct a dataset of over 15,000 images of Rome in just over 20 minutes.

Experiments have shown our method to be extremely accurate and efficient even at scale. Our method can be used on thousands of correspondences when the ground truth correspondences are known, or as a refinement step on inliers from a minimal estimation. However, as with all non-minimal pose solvers, it is difficult to make use of a large number of correspondences because of the likelihood of false features matches when using many

correspondences. For future work, we plan to explore ways to increase robustness to false correspondences by incorporating feature distances into the similarity transform estimation process so that our method can be more readily used with thousands of correspondences.

Part II

Calibrating the Distributed Camera

Chapter 5

Computing the Focal Length of a Single Camera

As discussed in Chapter 4, estimating the absolute camera pose from a set of 2D-3D correspondences, also known as the n-point pose (PnP) problem, is an important step in many Computer Vision applications such as Structure-from-Motion (SfM) [73], [89] and image-based localization [35, 52, 51, 69]. Especially for SfM, photo-community collections such as Flickr or Panoramio represent a vast and easily accessible source of data and truly enable large-scale 3D reconstructions [27]. Unfortunately, the EXIF data required to obtain the intrinsic camera calibration of the images is often missing for images obtained from photo sharing websites or is incorrect due to image editing operations applied before uploading the photos [10]. Thus, it is important to estimate both the camera pose and its internal calibration. For the latter, it is often sufficient to estimate only the focal length [12], [84].

Computing the camera pose for a calibrated camera is a well-understood problem that has been studied extensively [25], [31], [45], [49]. Given three correspondences between features in an image and points in the 3D model, the camera pose relative to the model can be computed very efficiently by solving a fourth degree polynomial [25], resulting in 3-point pose (P3P) solvers that require only about $2\mu s$ on a modern computer

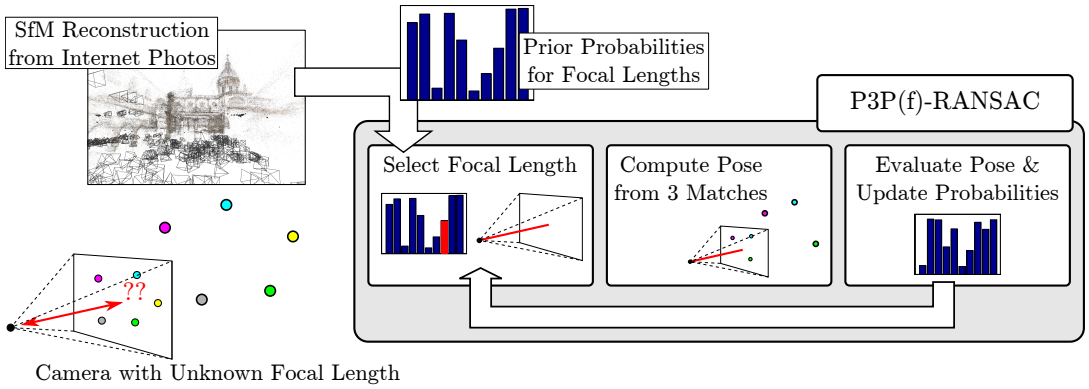

Figure 5.1: Illustration of the pose estimation strategy proposed in this chapter.

[45]. However, estimating the focal length together with the pose is a significantly harder problem. While special configurations such as planar scenes can be handled efficiently [1], computing both quantities generally requires solving a system of multivariate polynomials obtained from four or more 2D-3D correspondences [12], [84]. The bottleneck of such approaches is usually the Eigenvalue decomposition of the so-called action matrix [11], the resulting pose solvers require $46\mu s$ or more for a single instance. Consequently, using such methods inside a RANSAC-loop [25] results in prohibitively long run-times for all but high inlier ratios. In practice, it is thus common to employ pose solvers that achieve similar run-times as P3P [45] but require five or more 2D-3D correspondences [33],[46]. As the number of RANSAC iterations grows with both the percentage of false matches and the number of matches required to compute a pose, using such approaches results in a significantly larger number of RANSAC iterations, and thus higher run-times, compared to pose solvers using only three or four points for low inlier ratios (*cf*. Fig. 5.2(a)).

In this chapter, we consider the problem of estimating the camera pose for a camera with unknown focal length. Inspired by the brute-force approach of Irschara *et al*. [35], we propose to estimate the focal length by sampling from a discrete set of possible values, followed by computing the pose using the selected focal length instead of simultaneously estimating both quantities. As our main contribution, we propose a novel RANSAC

variant, called P3P(f)-RANSAC (*cf*. Fig. 5.1), that in each iteration randomly selects the focal length value based on the probability of finding a better model for it. In contrast to [35], which iteratively tests all possible focal length values, we re-estimate the probabilities of each possible focal length value after each RANSAC step using a recursive Bayesian filter. This enables our algorithm to quickly converge towards the focal length closest to the correct value. Consequently, our approach does not necessarily need to evaluate all focal length values, resulting in an average speed-up of more than one order of magnitude compared to [35]. We observe a distribution of focal lengths from photos obtained from photo-sharing websites that allow us to estimate the prior probabilities of the different focal length values, enabling our approach to use importance sampling to find a good pose more quickly. Through experiments on both large-scale SfM datasets and image-based localization tasks, we show that our proposed approach is significantly faster than the state-of-the-art minimal solver [12] while achieving a similar pose accuracy. At the same time, P3P(f)-RANSAC is faster than a recently published non-minimal solver [46] for low inlier ratios while achieving a higher localization accuracy[1].

The rest of the chapter is structured as follows. Sec. 5.1 discusses the problem solved in this chapter in more detail. We present our novel RANSAC variant combining probabilistic focal length sampling and pose estimation in Sec. 5.2. Sec. 5.3 then evaluates the resulting approach.

5.1 Problem Formulation

In this chapter, we want to solve the problem of estimating the pose for a camera with an unknown focal length from a given set $\mathcal{M} = \{(\mathbf{x}, \mathbf{X} \mid \mathbf{x} \in \mathbb{R}^2, \mathbf{X} \in \mathbb{R}^3\}$ of 2D-3D matches. Assuming that the principal point coincides with the center of the image, we

[1]We make our source code available at `http://people.inf.ethz.ch/sattlert`

are thus trying to determine the focal length $f \in \mathbb{R}$ and the rotation $\mathtt{R} \in \mathbb{R}^{3\times3}$ and translation $\mathbf{t} \in \mathbb{R}^3$ such that

$$\alpha \cdot \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} [\mathtt{R}|\mathbf{t}] \cdot \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix} \quad \text{for some scalar } \alpha > 0 \tag{5.1}$$

holds for all matches $(\mathbf{x}, \mathbf{X}) \in \mathcal{M}$, *i.e.*, that each 3D point $\mathbf{X}$ is projected onto its corresponding image position $\mathbf{x}$. In practice, some of the matches will be wrong due to imperfections in the matching process. The most common strategy to robustly handle wrong matches is to apply the PnP solver that computes the pose from n matches inside a RANSAC-loop [25] (*cf*. Section 2.2.2). Recall that assuming that each all-inlier sample allows us to estimate a high quality pose, the probability η of selecting an sample contaminated with at least one outlier (and thus a low quality pose) may be expressed as

$$(1 - \varepsilon^{*n})^k < \eta \ , \tag{5.2}$$

where k is the number of samples generated so far and ε^* is the *inlier ratio*, *i.e.*, the ratio of inliers among all matches, for the current best model. Thus, the maximal number of iterations required for a given inlier ratio ε is

$$k_{\max} = \log \eta / \log \left(1 - \varepsilon^n\right) \ . \tag{5.3}$$

The probability of selecting an all-inlier sample is maximized by minimizing n. However, the minimal 4-point solver (P4Pf) [11] for the problem of estimating both the pose and the focal length requires $46\mu s$, which is prohibitively expensive for low inlier ratios where many RANSAC iterations are required. Faster pose solvers such as the P5Pfr

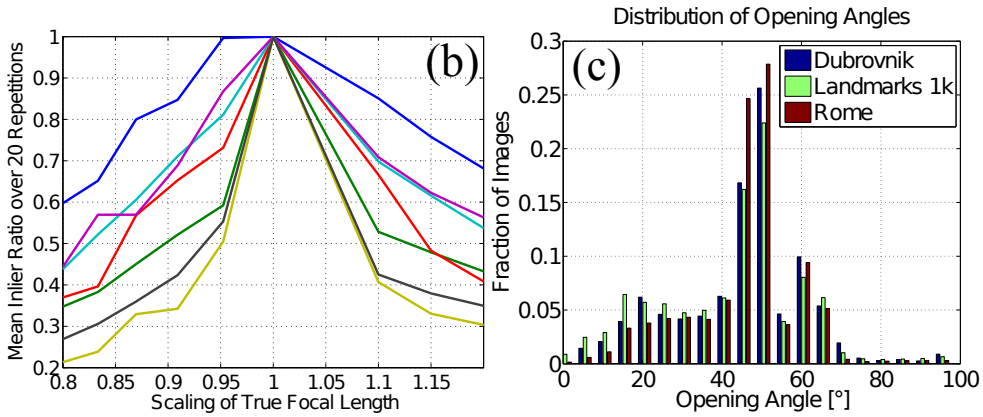

Figure 5.2: (a) The focal length accuracy required to recover most of the inliers strongly varies between different cameras. Yet, the inlier ratio decreases monotonically on both sides of the optimal focal length value. (b) Histograms of opening angles from images in the Dubrovnik [50], Landmarks 1k [51], and Rome [52] datasets.

method [46] that estimates the pose, focal length, and radial distortion of the camera from five matches exist. However, using a non-minimal n reduces the probability of selecting an all-inlier sample exponentially, resulting in a significant increase in the number of required iterations for low inlier ratios (*cf*. Fig. 2.4 right). Instead of using a non-minimal solver, we propose to use a 3-point solver that estimates the pose for a given focal length f [45] and select f from a pre-defined set $\mathcal{F}$ of focal length values. This strategy offers the possible advantage of requiring fewer iterations than RANSAC with P4Pf and faster pose computation times by using the P3P solver.

Evaluating all focal length values in $\mathcal{F}$ independently from each other as proposed by [35] will require at least $|\mathcal{F}| \cdot k_{\max}(f_{\text{gt}})$ iterations in total, where $k_{\max}(f_{\text{gt}})$ is the maximum number of iterations required to confidently compute the pose when using the ground truth focal length. Consequently, the approach from [35] will only be more efficient than using RANSAC with P4Pf or P5Pfr if $|\mathcal{F}|$ is smaller than the difference in the pose solver time or the difference in the number of required iterations, respectively. Notice that using quantized focal length values will invariably result in a lower pose accuracy. Regardless, as long as we are able to recover most of the inliers we will be able to obtain a better pose

Algorithm 1 P3P(f)-RANSAC

Given: Set $\mathcal{M}$ of 2D-3D matches, confidence threshold η, **set $\mathcal{F}$ of focal length values with prior probabilities $P_{\text{prior}}(f)$ for all $f \in \mathcal{F}$**

initialize sampling probability $P_{\text{sample}}(f) = P_{\text{prior}}(f)$ **for all** $f \in \mathcal{F}$
while probability of having missed all-inlier sample $\geq \eta$ **do**
  randomly select focal length $f \in \mathcal{F}$ **according to** P_{sample}
  draw random sample $s \subset M$ of size 3
  estimate pose $[\mathtt{R}|\mathbf{t}]$ from s with a P3P solver **using** f
  evaluate pose hypothesis $\theta = (f, [\mathtt{R}|\mathbf{t}])$ on $\mathcal{F}$
  if new best model found **then**
    $\theta^* = (f, [\mathtt{R}|\mathbf{t}])$
  Update probabilities P_{sample}
  Re-estimate probability of having missed an all-inlier sample

Return: θ^*

by applying P4Pf on the resulting inliers with only a small run-time overhead as very few sampling steps will be needed. Unfortunately, the sampling density required to guarantee that we can select a focal length value close enough to f_{gt} to recover most of the inliers strongly depends on the depth-variation of the scene observed by the camera. This can be seen in Fig. 5.2(a), as we observe different sensitivities on the focal length accuracy for different cameras. Thus, we need a rather dense sampling in order to handle all types of scenes, resulting in a large set $\mathcal{F}$. In order to maintain fast run-times when using a large set of values, we model the dependencies between the different focal lengths, enabling us to avoid evaluating all focal length values for at least $k_{\max}(\varepsilon_{\text{gt}})$ steps. This can be done by exploiting a *key observation* that can be made from Fig. 5.2(a): The maximal inlier ratio obtained by RANSAC for each focal length value decreases monotonically with the distance to f_{gt}. Given the focal length used to generate the current pose estimation with the highest inlier count, f^*, this observation allows us to model the probability of finding a pose with a higher inlier ratio using another focal length f as a function of $|f - f^*|$.

5.2 Probabilistic Focal Length Sampling

The main idea of our novel pose estimation approach is to use focal length sampling and a P3P solver [45] in order to estimate a hypothesis for the camera pose from $n=3$ 2D-3D correspondences instead of computing the pose and focal length simultaneously from four matches or more. Once we have found a good pose with a high inlier ratio for a focal length f^*, it becomes very unlikely that focal length values f far away from f^* can be used to estimate a better pose (*cf.* Fig 5.2(a)). The central idea behind our approach is thus to preferably select focal length values that have a high likelihood of yielding a pose with a larger number of inliers than the current best estimate. This naturally leads to a probabilistic formulation of the problem of selecting good focal length values. This probabilistic formulation in turn enables us to exploit the fact that certain focal length values are much more likely to be correct than others. Alg. 1 outlines the resulting RANSAC variant, where differences to the classical RANSAC algorithm [25] are highlighted. Besides the 2D-3D matches and a confidence threshold, our approach requires a set $\mathcal{F}$ of focal length values with associated prior probabilities as an additional input. These priors are then used to initialize the probability distribution that we use for selecting the focal length value f in Line 3 of Alg. 1. After using P3P to generate a pose hypothesis from f and three randomly selected matches, the hypothesis is evaluated on all matches and the current best pose estimate is updated if necessary. Finally, we use a recursive bayesian filter to re-estimate the probability distribution used for sampling the focal length to reflect the fact that the current iteration might influence the likelihood of finding a better pose for all other focal length values.

In the following, we will refer to our algorithm as P3P(f)-RANSAC, as it uses a P3P solver inside of a RANSAC loop, where the focal length value f is obtained via parameter sampling. Similarly, we will refer to RANSAC-loops using any other PnP solver as PnP-RANSAC.

In Sec. 5.2.1, we briefly explain how to obtain the prior probabilities for the focal

length values from $\mathcal{F}$. As the main contribution of this chapter, Sec. 5.2.2 derives the probability distribution used for sampling the focal length values and our strategy for re-estimating the sampling probabilities. Finally, Sec. 5.2.3 argues that using early model rejection techniques [16], [18] is crucial for our RANSAC variant in order to offer faster run-times than P4Pf and P5Pfr.

5.2.1 Obtaining the Prior Probabilities

The focal length of a camera mainly depends on the type of camera and the zoom-level used to take the picture. In this chapter, we consider pose estimation scenarios in which a large variety of camera types is used, as is the case in large-scale SfM reconstructions from images downloaded from Flickr [27], [73]. Since some camera types are much more popular than others[2], not all focal length values are equally likely to occur. The cameras contained in a large-scale SfM reconstruction of community collection photos thus give us an approximation to the probability distribution of focal length values. However, notice that obtaining prior probabilities for focal length values is an ill-posed problem as the focal length depends on the image resolution. In contrast, the maximal opening angle $\alpha_{\max}$ of a camera with focal length f, width w, and height h, related by

$$\tan\left(\alpha_{\max}/2\right) = \frac{\max\left(w,h\right)}{2 \cdot f} \quad , \tag{5.4}$$

is independent of the image resolution. Thus, we predetermine a set of opening angle values from cameras contained in large-scale SfM reconstructions of unordered image collections [52],[51]. We transform the opening angles to focal length values via Eqn. 5.4 before applying P3P(f)-RANSAC based on the resolution of the image being localized. Fig. 5.2(b) shows the distribution of opening angles for three such datasets, Dubronik

[2] `https://www.flickr.com/cameras`

(6k images) [52], Rome (15k images) [52], and the Landmarks 1k dataset (205k images) [51]. The distribution of opening angles is consistent across all datasets, indicating that the resulting distributions are a good representation of images taken in the real world. However, we will show in Sec. 5.3.2 that the choice of priors is not a crucial parameter.

5.2.2 Obtaining and Re-estimating the Sampling Probabilities

Ideally, the probability $P_{\text{sample}}(f)$ of selecting a focal length f should be proportional to the likelihood of obtaining a pose estimate with an inlier ratio $\varepsilon(f)$ that is larger than the inlier ratio ε^* of the current best pose estimate θ^* obtained for focal length f^*. Consequently, we model the sampling probability as

$$P_{\text{sampling}}(f) = \frac{P(\varepsilon(f) > \varepsilon^* \mid f) \cdot P_{\text{prior}}(f)}{\sum_{f' \in \mathcal{F}} P(\varepsilon(f') > \varepsilon^* \mid f') \cdot P_{\text{prior}}(f')} \ , \tag{5.5}$$

where $P(\varepsilon(f) > \varepsilon^* \mid f)$ is the probability of finding a better model for f. As is common in practice [52], [69], we assume that we can obtain an inlier ratio of at least ε_0 in order to limit the maximal number of RANSAC iterations, *i.e.*, we assume $\varepsilon^* = \varepsilon_0$ until we find a pose with an inlier ratio $> \varepsilon_0$.

In the following, we first derive $P(\varepsilon(f) > \varepsilon_0 \mid f)$ for the case that all models found so far were computed from samples containing at least one outlier. In this case, we haven't found a good model so far and thus have to treat all focal length values independently. We then show that the case of having found a good model with $\varepsilon^* > \varepsilon_0$, in which case $P(\varepsilon(f) > \varepsilon^* \mid f)$ depends on the current best pose θ^*, seamlessly integrates into our definition of the probabilities.

Case 1: $\varepsilon^* = \varepsilon_0$. Using RANSAC's termination criterion from Eqn. 5.2, we can express the maximal inlier ratio $\varepsilon_{\max}(f)$ that we can find with certainty $\geq \eta$ in terms of the

number of random samples $k(f)$ generated so far for focal length f:

$$\varepsilon_{\max}(f) = \sqrt[3]{1 - \sqrt[k(f)]{\eta}} \ . \tag{5.6}$$

Since we are only required to compute the correct pose with certainty greater than η, the probability $P(\varepsilon(f) > \varepsilon_0 \mid f)$ of finding a model with a higher inlier ratio is directly related to the probability that the number of correct matches found in $\mathcal{M}$ is in the range $(\varepsilon_0 \cdot |\mathcal{M}|, \varepsilon_{\max}(f) \cdot |\mathcal{M}|]$. Notice that the probability of finding a wrong match only depends on the matching algorithm and the structure of the 3D model [69], and *not* on the pose estimation strategy itself. Since this probability can be estimated empirically from training data, we can assume without loss of generality that we know the cumulative distribution function $\mathrm{cdf}(\varepsilon)$ over the inlier ratios for the given matching algorithm and 3D model. Thus, we can express the probability of finding a better model for f as

$$P(\varepsilon(f) > \varepsilon_0 \mid f) = \mathrm{cdf}(\max(\varepsilon_{\max}(f), \varepsilon_0)) - \mathrm{cdf}(\varepsilon_0) \ . \tag{5.7}$$

Under the reasonable assumption that $\mathrm{cdf}(\varepsilon)$ is strictly increasing, *i.e.*, that all inlier ratios occur with a non-zero probability, we have $P(\varepsilon(f) > \varepsilon_0 \mid f) = 0$ only if $\varepsilon_{\max}(f)) \leq \varepsilon_0$. Consequently, P3P(f)-RANSAC will terminate after $|\mathcal{F}| \cdot k_{\max}(\varepsilon_0)$ iterations, *i.e.*, if no pose with inlier ratio greater than ε_0 can be found with an uncertainty below η.

Case 2: $\varepsilon^* > \varepsilon_0$. Note that $P(\varepsilon(f) > \varepsilon^* \mid f)$ not only depends on the inlier ratio ε^* but also on the value of the focal length f^* used to compute the current best hypothesis θ^*. If f^* is close to the correct focal length f_{gt}, then focal length values far away from f^* are much less likely to result in better pose hypotheses than values close to f^*. This behavior can also be observed in Fig. 5.2(a), which shows that the inlier ratio decreases monotonically with the distance to the correct focal length when applying RANSAC on

correct matches only. While outlier matches might cause small local maxima, we found that this relation is still a very good model in practice. Since a similar behavior has been observed for other estimation problems [59], we thus use the following simplifying assumption to derive the sampling probabilities.

Assumption 1 *Let $\varepsilon(f)$ be the maximal inlier ratio that can be obtained for focal length f and let f_{gt} be the correct focal length. For focal length values f and f' with $|f_{gt} - f'| < |f_{gt} - f|$, $\varepsilon(f) \leq \varepsilon(f') \leq \varepsilon(f_{gt})$ should hold.*

Without loss of generality, consider the focal length $f < f^*$. If f is closer to f_{gt} than f^*, Assumption 1 implies that we should be able to find an inlier ratio of at least ε^* for all $f' \in \mathcal{F} \cap [f, f^*)$. Let $\mathcal{F}(f, f^*) = \mathcal{F} \cap [f, f^*)$ be the set of corresponding focal length values and let $P(\varepsilon(\mathcal{F}(f, f^*)) > \varepsilon^* \mid f)$ denote the probability of finding a better pose in the range $[f, f^*)$, then we have

$$P(\varepsilon(f) > \varepsilon^* \mid f) \quad \leq \quad P(\varepsilon(\mathcal{F}(f, f^*)) > \varepsilon^* \mid f) \ . \tag{5.8}$$

The maximal inlier ratio expected in this range of focal lengths with a certainty of at least η is again given by

$$\varepsilon_{\max}(\mathcal{F}(f, f^*)) = \sqrt[3]{1 - \sqrt[k(\mathcal{F}(f,f^*))]{\eta}} \ , \tag{5.9}$$

where $k(\mathcal{F}(f, f^*)) = \sum_{f' \in \mathcal{F}(f,f^*)} k(f')$ is the sum over all samples generated for the focal length from the considered range. Following the same argumentation as in Case 1, we obtain

$$P(\varepsilon(f) > \varepsilon^* \mid f) = \text{cdf}(\max(\varepsilon_{\max}(\mathcal{F}(f, f^*)), \varepsilon^*)) - \text{cdf}(\varepsilon^*) \ . \tag{5.10}$$

This predict-and-update strategy is a recursive bayesian filter. Note that we again have $P(\varepsilon(f) > \varepsilon^* \mid f) = 0$ only if the probability of finding a better pose for f drops above

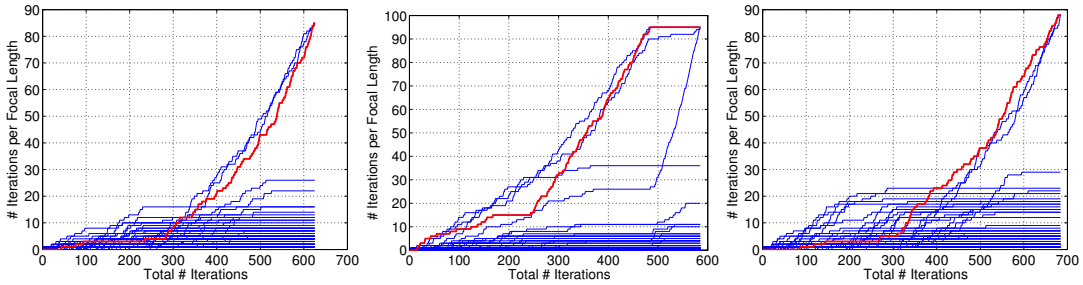

Figure 5.3: The number of iterations in which each of the 100 focal length values is selected, plotted over the iterations of P3P(f)-RANSAC for three cameras from the Dubrovnik dataset and an outlier ratio of 50%. The focal length value closest to the true focal length of each camera is highlighted in red. As can be seen, P3P(f)-RANSAC is able to quickly identify a subset of promising focal lengths while ignoring all other values.

the confidence threshold η, *i.e.*, P3P(f)-RANSAC essentially uses the same termination criterion as original RANSAC, offering the same guarantees on the quality of the pose.

Behavior of the proposed sampling strategy. As long as no pose with an inlier ratio above ε_0 is found (Case 1), P3P(f)-RANSAC essentially uses importance sampling to select promising focal length values. As soon as a good model with inlier ratio above ε_0 is found (Case 2), P3P(f)-RANSAC is able to model the dependencies between focal length values, allowing it to quickly focus on a smaller range of focal length values that are most likely to be correct. This behavior is illustrated in Fig. 5.3. At the same time, our sampling strategy is able to escape local maxima since all focal length values that could lead to a better pose have a non-zero probability of being selected.

Implementation details. Each focal length value is used for at most $k_{\max}(\varepsilon_0)$ samples. Since both Eqn. 5.6 and Eqn. 5.9 only depend on the number of iterations and not on ε^*, we can use a lookup table to determine the maximal inlier ratio. We represent the (empirically determine) cumulative distribution function $\mathrm{cdf}(\varepsilon)$ as a discrete set of values. For any inlier ratio ε', we use linear interpolation to compute $\mathrm{cdf}(\varepsilon')$. This guarantees $\mathrm{cdf}(\varepsilon') > \mathrm{cdf}(\varepsilon^*)$ if $\mathrm{cdf}(\varepsilon)$ is strictly increasing and thus prevents P3P(f)-RANSAC from

terminating too early.

5.2.3 Integrating Early Model Rejection

The P3P solver can compute the pose from three 2D-3D matches in $2\mu s$ [45] while the fastest P4Pf solver takes $46\mu s$ [11]. Consequently, P3P(f)-RANSAC should be able to perform 23 times more sampling steps while still being faster than P4Pf-RANSAC. However, evaluating the computed pose on the set of matches also has a significant impact on the run-time of a single RANSAC iteration. Since evaluating a pose takes around $20 - 50\mu s$ (or more for images with a large number of matches), P3P(f)-RANSAC can be at most $2 - 3$ times faster than P4Pf-RANSAC when evaluating each pose on all matches. Obviously, we do not need to fully evaluate poses generated from non-all-inlier samples or with a wrong focal length value. We can thus use approaches that terminate the pose evaluation once it becomes likely that the current pose will not have an inlier ratio higher than ε^* [16, 18]. We chose to use the simple $T_{d,d}$ test, which evaluates a pose on all matches only if d randomly selected matches are inlier to the pose [16], with $d=1$ as proposed in [16]. As a result of applying this $T_{1,1}$ test, we need to draw $n=4$ matches in each iteration of P3P(f)-RANSAC, increasing the number of required iterations (*cf.* Eqn. 5.3). At the same time, it becomes rather unlikely that any pose estimated from a focal length far away from the correct value, even if it was estimated only from correct matches, is evaluated on all correspondences as significantly fewer correct matches are inliers to such poses (*cf.* Fig. 5.2(a)). As a consequence, only a small fraction of all generated poses need to be fully estimated, resulting in a significant speed-up.

5.3 Image-based Localization Evaluation

In the following, we evaluate the performance of our proposed method both on synthetic and real-world data. For all experiments, we use the Landmarks 1k dataset [51], reconstructed from 205k Flickr images, to learn the probability distribution for 100 equally spaced opening angles, which we then transform into focal length values for any image with a given width and height.

Using realistic focal lengths is an important part of our experiments, since our algorithm utilizes the distribution of likely focal lengths to inform our RANSAC scheme. In order to obtain realistic focal length values, and realistic 2D-3D matches, for our synthetic experiments, we use two large-scale SfM reconstructions and generate pixel-perfect 2D-3D correspondences by reprojecting the 3D points into the images in which they were observed. The Rome model [52] consists of 15k database images and 4M points, while 1.9M points were reconstructed from 6k images to create the Dubrovnik model [52]. The scale for the latter model is known, allowing us to measure the localization accuracy on the Dubrovnik dataset in meters. Both datasets are form a standard benchmark for image-based localization tasks [51], [69] and we thus evaluate the performance on real-world data of our approach in this application scenario. For both datasets we use a CDF learned from inlier ratios observed on the Dubrovnik dataset.

For our experiments, we used the publicly available implementations of the P3P [45] and P4Pf [12] and our own implementation of the P5Pfr solver [46].

5.3.1 Experiments with Synthetic Data

We conducted two synthetic experiments to measure the performance of our algorithm under increased levels of image noise and outlier ratios.

Image noise. We measured our algorithm's robustness against image noise by adding

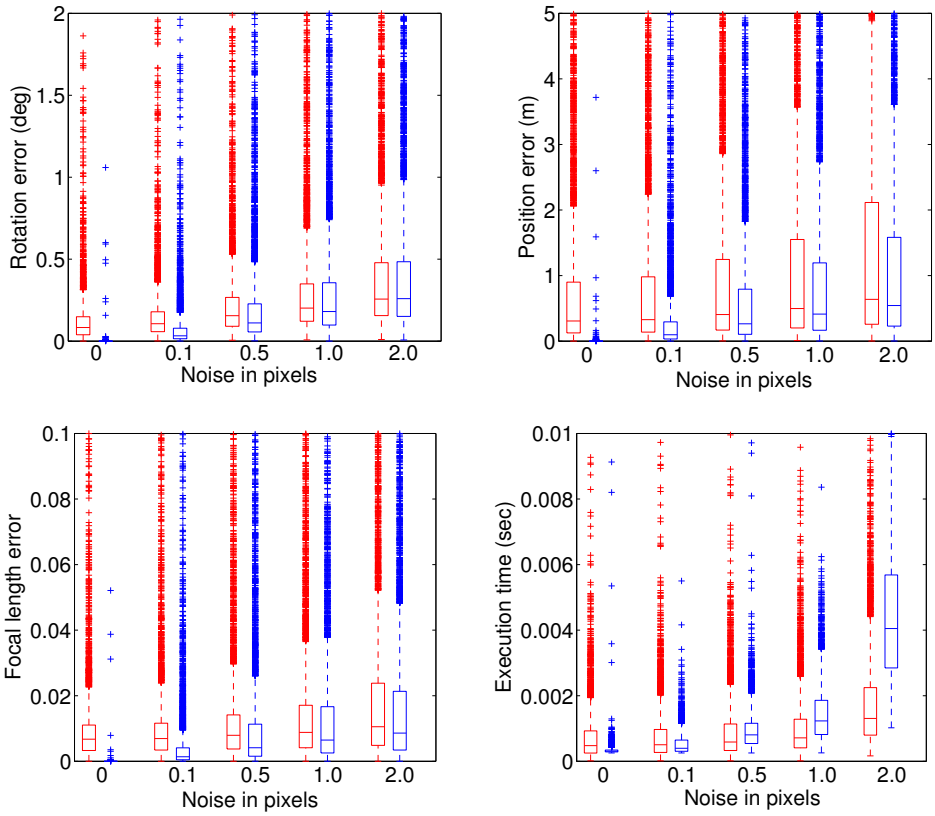

Figure 5.4: Performance of our algorithm (red) and P4Pf [12] (blue) are compared for increased levels of image noise. Our algorithm has comparable performance to P4Pf for rotation, position, and focal length errors for all levels of noise. Despite requiring more iterations, our algorithm has a lower run-time than P4Pf as the image noise increases.

increasing levels of Gaussian pixel noise to the 2D positions of the perfect 2D-3D correspondences obtained by reprojecting the 3D points. We tested image noise levels of 0, 0.1, 0.5, 1.0, and 2.0 pixels. Fig. 5.4 compares the performance of our approach with P4Pf-RANSAC. For all levels of image noise, P4Pf achieves slightly lower rotation, translation, and focal length errors, though the errors are comparable. This indicates that our algorithm is able to estimate the pose and focal length with high precision and is thus robust to noise, which is important for real-world data.

Outlier ratio. The key idea of our approach is to use the faster P3P solver to estimate

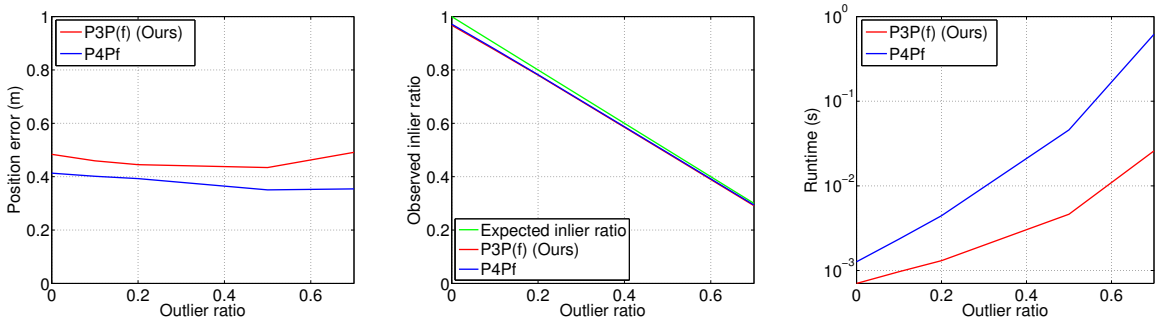

Figure 5.5: The median position error, inlier ratio, and run-time was measured while increasing the outlier ratio from 0 to 0.7. Both algorithms are able to recover high quality poses (left) and almost all expected inliers (middle). Our algorithm has a much lower run-time than P4Pf (right) as the outlier ratio increases because we require fewer correspondences. This is a major advantage of our algorithm in low-inlier scenarios.

camera poses more efficiently while avoiding a brute-force search through all possible focal length values through our novel sampling scheme. In this experiment, we evaluate the robustness of our approach to high outlier ratios. We again use the perfect matches from the Dubrovnik dataset, with 1 pixel of Gaussian noise added to the reprojected points, and create outliers by adding new image points with correspondences to 3D points that were not observed in the image until the desired outlier ratio is achieved.

Fig. 5.5 shows the performance of our P3P(f) approach and P4Pf-RANSAC for increasing levels of outlier ratios. We plot the median position errors, inlier ratios, and execution times. As can be seen, our algorithm is able to handle low-inlier scenarios and still produce results that are nearly as accurate as P4Pf while being several orders of magnitude faster. These results demonstrate that Assumption 1 holds well enough even in the presence of outliers. For tasks such as image-based localization, being able to handle low-inlier scenarios accurately and efficiently is extremely important.

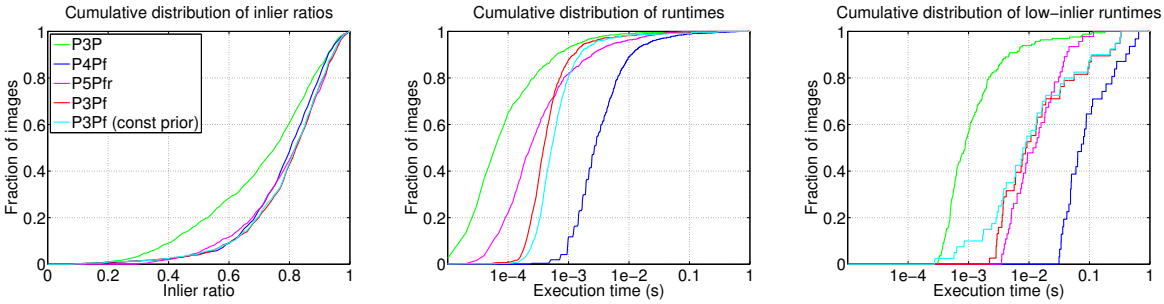

Figure 5.6: Localization results from the Rome dataset [52] are shown. Our P3P(f)-RANSAC algorithm is able to recover more inliers than P3P used with ground truth focal lengths from Bundler, and a comparable amount to P4Pf and P5Pfr (left). Our algorithm has an execution time that is nearly one order of magnitude faster than P4Pf (center), despite running for more iterations. In low-inlier cases (inlier ratio ≤ 0.5), our algorithm is significantly faster than alternative algorithms (right).

5.3.2 Experiments on Real Data

As a final experiment, we compare the performance of our algorithm to P3P, P4Pf, and P5Pfr in an image-based localization task [52, 51], [69]. We use two versions of our algorithm: one with priors obtained from the Landmarks 1K dataset, and one with no priors (*i.e.* constant priors). We use the efficient, publicly available localization method of [69] to obtain 2D-3D matches for the 800 and 1000 query images available for the Dubrovnik and Rome datasets, respectively. All query images were obtained by removing cameras from larger SfM reconstruction, providing ground truth positions for the query images. Notice that we do not use perfect correspondences in these experiments.

The results for the Rome dataset are shown in Fig. 5.6. Algorithms that computed focal length in addition to pose are able to recover noticeably more inliers than the P3P method that was used with ground truth focal lengths values as we did not account for radial distortion. As expected, all of the algorithms are slower than P3P. Our algorithm performed much faster than P4Pf in all cases. As shown in Fig. 5.6, our approach is faster than P5Pfr for low-inlier cases since it requires fewer matches per sample and thus fewer iterations per focal length.

Tab. 5.1 shows the position errors of each method on the Dubrovnik dataset, where we can measure distances in meters. The median position error of each camera was recorded over 100 trials for each of the methods. All methods are able to localize almost all images, and our method gives position errors that are comparable to or only slightly higher than P4Pf, which has the lowest errors of all algorithms. P3P(f) achieves better localization accuracy than P5Pfr. As can be also seen in Tab. 5.1, our method is on average over an order of magnitude faster than P4Pf. At the same time, P3P(f) is consistently faster than P4Pf on all quantiles while being faster than P5Pfr for images with lower inlier ratios. Notice that our P4Pf implementation requires $115\mu s$ compared to the $46\mu s$ required by [12]. Yet, our approach is on average more than 7 times faster than when using the solver from [12] and still achieves faster quantile run-times. On average, P3P(f) is only 1.39 times slower than P3P, even though it requires no knowledge about the focal length, making it well suited for SfM and localization applications.

Tab. 5.1 also shows results obtained with our approach when using a uniform prior on the focal lengths instead of the learned one. As can be seen, our method benefits from using a realistic prior but performs only slightly worse using the uniform prior. This demonstrates that our novel sampling scheme is the main reason for why P3P(f) succeeds.

5.4 Discussion

In this chapter, we have proposed a novel approach, termed P3P(f)-RANSAC, for efficiently estimating the pose of a camera with unknown focal length inside a RANSAC loop. Instead of computing the focal length using a minimal solver, our approach samples focal length values according to a probability distribution and then uses the significantly faster P3P solver to estimate the pose of the now calibrated camera. As the main contri-

Table 5.1: The position errors and localization times measured on the Dubrovnik dataset for an image-based localization task. Besides the results obtained by our approach using the learnt priors for the focal lengths, we also include results for an uniform prior

Solver	# loc. images	Localization Accuracy [m]				
		Mean [m]	Quantiles [m]			
			25%	50%	75%	90%
P3P (exact focal)	792	40.3	1.0	7.6	26.4	111.8
P4Pf	795	38.7	**0.4**	**1.3**	**4.7**	**20.1**
P5Pfr	**796**	227.2	0.5	2.0	31.3	200.9
P3P(f) (Ours)	795	**20.8**	**0.4**	1.6	5.4	27.6
P3P(f) uniform prior	795	28.1	0.5	1.7	5.9	24.3

Table 5.2: The execution time required for each method in the localization experiment on the Dubrovnik dataset. Our method, when used with or without priors, is nearly as efficient as P3P with known calibration.

Solver	# loc. images	Localization Times [ms]			
		Mean [ms]	Quantiles [ms]		
			50%	75%	90%
P3P (exact focal)	792	**1.21**	**0.20**	**1.00**	3.01
P4Pf	795	32.09	4.84	10.78	28.73
P5Pfr	**796**	6.02	0.54	3.07	16.44
P3P(f) (Ours)	795	1.68	0.68	1.27	**2.72**
P3P(f) uniform prior	795	1.89	0.85	1.46	3.08

bution, we have proposed a novel sampling scheme that is able to model the probability of finding a pose better than the current best estimate for all focal length values. As a consequence, our approach is able to avoid evaluating all values and focus on the more promising candidates while offering the same guarantees as RANSAC in the presence of outliers. We have shown that our algorithm achieves a similar pose accuracy as previous pose solvers while achieving significantly faster run-times. These results challenge the notion that using minimal solvers is always an optimal strategy. While this chapter focuses on the absolute pose problem, we hope to explore the use of our framework for

other pose estimation problems in future work.

Chapter 6

Computing Camera Focal Lengths at Scale

A current limitation of global SfM methods is that they require relative poses in the form of relative rotations and translations as input. For calibrated image sets, the relative poses may be obtained by decomposing the essential matrix [33]. For uncalibrated cameras, only the fundamental matrix is available between two views. Focal lengths may be obtained from the fundamental matrix in closed form [40] and the resulting essential matrix may be decomposed into relative rotations and translations. The relative rotations and translations obtained through fundamental matrix decomposition, however, are far less accurate compared to when calibration is known (*cf*. Figure 6.3) so obtaining accurate calibration has a direct effect on the quality of SfM algorithms.

Individually decomposing fundamental matrices from all relative geometries containing a particular camera, however, is not guaranteed to yield a single consistent focal length value. That is, each decomposition of a fundamental matrix containing a particular camera may yield a different focal length value for that camera (*cf*. Figure 6.2). Further, the quality of the focal lengths computed from a fundamental matrix is solely dependent on the quality of the fundamental matrix estimation. Focal lengths are not a lie group and so a simple averaging of focal lengths does not give statistically meaningful

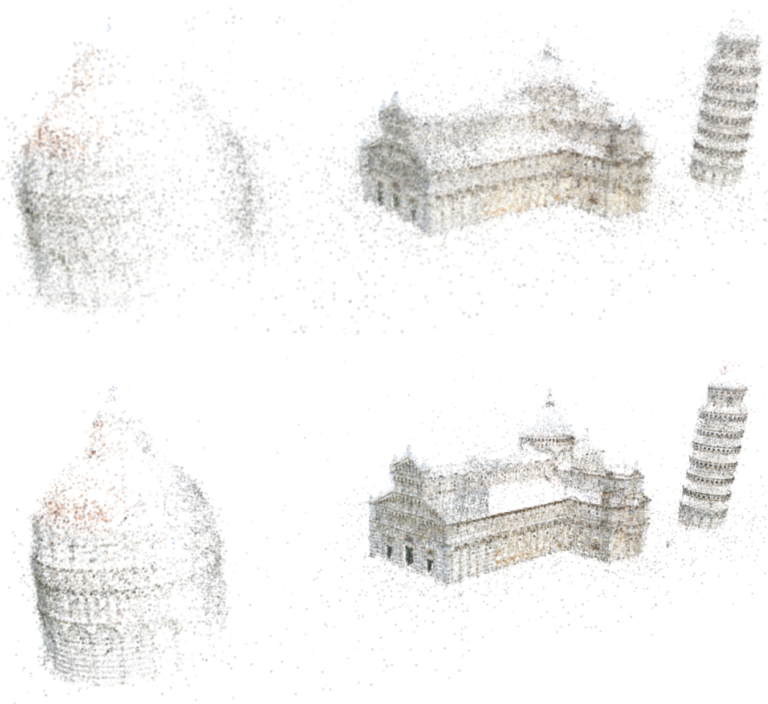

Figure 6.1: Reconstructions computed from global SfM methods on the Pisa dataset [38]. **Top:** Standard global SfM pipelines [88] struggle to handle image sets with poor calibration or inaccurate relative geometries, resulting in noisy or disconnected reconstructions. **Bottom:** Our method optimizes the relative geometries in the viewing graph to enforce global consistency, resulting in an efficient SfM pipeline that handles calibrated or uncalibrated images.

results [13] and a more meaningful metric is needed to effectively "average" focal lengths. In this section we propose a new calibration method for simultaneously determining the focal lengths of all cameras in a viewing graph using only fundamental matrices as input.

6.1 Focal Lengths from a Fundamental Matrix

First, let us review a technique for determining focal lengths from a single fundamental matrix. An essential matrix E has the form $t \times R$ for a given relative translation t and rotation R if and only if E is rank 2 with its two non-zero singular values equal [33].

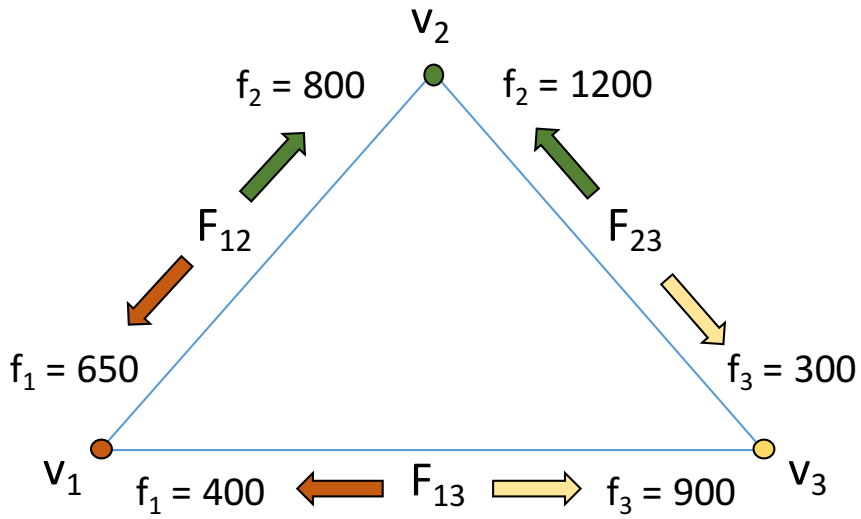

Figure 6.2: Camera focal lengths may be computed by decomposing the fundamental matrix connecting two views; however, when multiple fundamental matrices are connected to a view, multiple estimates for the focal length of the view may exist leading to inconsistent calibration.

This property may be encapsulated by the scalar invariants of E [39]:

$$C = ||EE^\top||^2 - \frac{1}{2}||E||^4 \ . \tag{6.1}$$

For a valid essential matrix E, the cost function C will be 0. Kanatani and Matsunaga [40] show that Eq. (6.1) may be used to recover the two focal lengths from a fundamental matrix by noting that:

$$E = K'^\top F K \ . \tag{6.2}$$

When the focal lengths are unknown, C is a non-negative cost function whose minimum is at 0. By inserting Eq. (6.2) into Eq. (6.1), we may solve for the focal length values that minimize C. This may be solved in closed form by noting that the first order partial derivatives $\partial C/\partial f'$ and $\partial C/\partial f$ must also be 0 [40].

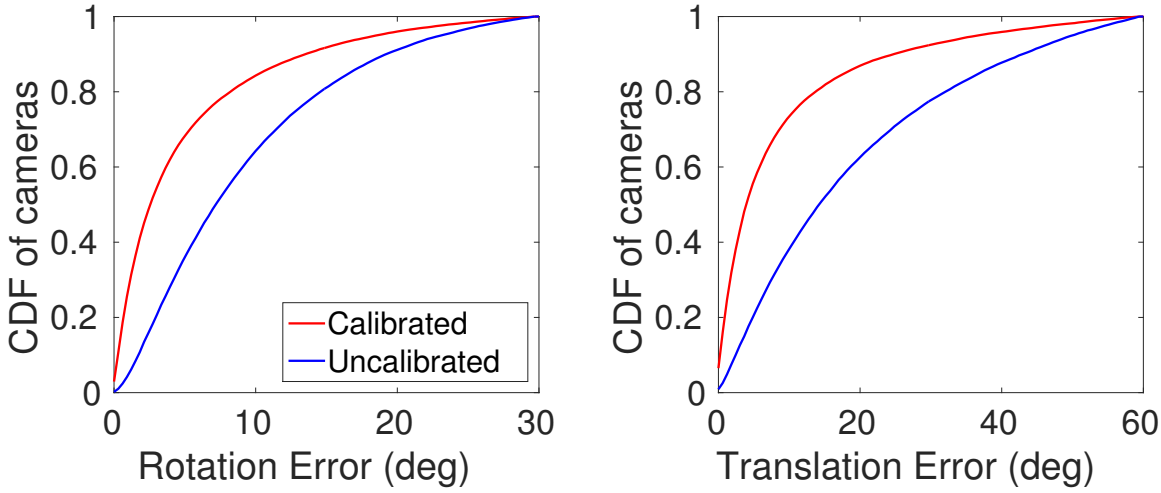

Figure 6.3: We measured the effect of calibration on relative pose error. When using known calibration (red) the relative rotation and translations are significantly more accurate then when calibration is unknown (blue). For unknown calibrations, we compute relative rotations and translations by decomposing the fundamental matrix.

6.2 Focal Lengths from the Viewing Graph

Kanazawa *et al.* [41] extend Eq. (6.1) to a triplet of fundamental matrices with a simple cost function:

$$C = C(F_{12}) + C(F_{13}) + C(F_{23}) \ . \tag{6.3}$$

When image noise is present, this non-negative cost function is no longer guaranteed to have a minimum at $C = 0$; however, minimizing this function is shown to produce good estimations of focal lengths for the triplet [41]. We extend this triplet formulation to operate on an entire viewing graph:

$$\boldsymbol{f}^* = \arg\min \sum_{F \in \mathcal{G}} C(F) \ , \tag{6.4}$$

where $\boldsymbol{f}^* = \{f_0, \ldots, f_n\}$ is the set of all focal lengths of all views in the viewing graph $\mathcal{G}$. The focal length values are obtained by minimizing the cost function of Eq. (6.1) over

all fundamental matrices that correspond to edges in the viewing graph. We use an L_1 loss function to minimize the terms of Eq. (6.4) to maintain robustness to outliers.

The minimization of Eq. (6.4) can easily be modified to handle viewing graphs with partially known calibration by keeping the known focal lengths constant during the minimization. Similarly, Eq. (6.4) can be easily modified to handle the case of all cameras sharing the same focal length.

6.3 The Viewing Graph

A scene consisting of n views may be represented by a *viewing graph* $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ whose vertices $\mathcal{V}$ correspond to views in the scene and whose edges $\mathcal{E}$ correspond to feature matches and relative geometries between two views, namely the fundamental matrix connecting two views. Specifically, F_{ij} is the fundamental matrix that transfers points in image j to lines in image i. The viewing graph contains information about the relative geometry between views but does nothing to enforce geometric constraints beyond 2-view geometry. For example, there may be triplets (loops of size 3) whose relative geometry is not geometrically feasible when considering all three edges [68, 91]. Ideally, the edges in these loops would be consistent with each other.

Condition 1 *A triplet of fundamental matrices is consistent when they satisfy [33]:*

$$e_{ik}^\top F_{ij} e_{jk} = e_{ij}^\top F_{ik} e_{kj} = e_{ji}^\top F_{jk} e_{ki} = 0 \ , \tag{6.5}$$

where e_{ij} is the epipole of F_{ij} corresponding to the image of camera center j in view i and $e_{ij} \neq e_{ik}$ i.e.*, the non-collinearity condition is satisfied.*

Definition 1 *A consistent viewing graph is a viewing graph where all triplets satisfy Condition 1.*

The geometric interpretation of Definition 1 is that the projection of view k's camera center in image i is consistent with the projection of view k's camera center in image j transferred to image i by the fundamental matrix F_{ij}.

Let us now consider the existence of a consistent viewing graph:

Theorem 1 *Given a reconstruction $\mathcal{R} = \{\mathcal{P}, \mathcal{X}\}$ consisting of projection matrices $\mathcal{P}$ and 3D points $\mathcal{X}$, a non-empty set of consistent viewing graphs exists.*

Proof: A consistent viewing graph may be constructed directly from the reconstruction $\mathcal{R}$ by setting each edge $e_C \in \mathcal{E}$ to the fundamental matrix composed from the two corresponding projection matrices [33]. By construction, Condition 1 is satisfied. ■

Thus, for every reconstruction $\mathcal{R}$ there exists a consistent viewing graph $\mathcal{G}_C$ that will generate $\mathcal{R}$. Further, it is known that computing a reconstruction from a consistent viewing graph may be done trivially by chaining projection matrices computed directly from the fundamental matrices in the viewing graph [68, 71]. Computing a reconstruction from a non-consistent viewing graph, however, is much more difficult and is the crux of most SfM methods.

6.4 Creating a Consistent Viewing Graph

Rather than facing the difficult task of computing a reconstruction from a non-consistent viewing graph $\mathcal{G}$, we propose to instead recover a consistent viewing graph $\mathcal{G}_C$ from $\mathcal{G}$ so that computing a reconstruction is simplified [33, 68]. Thus, the goal of this chapter is to optimize a noisy, non-consistent viewing graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ to recover a consistent viewing graph $\mathcal{G}_C$ that will improve SfM. This requires adjusting the edges $F_{ij} \in \mathcal{E}$ to enforce Condition 1. We propose an optimization scheme that uses a geometric error to enforce loop constraints that attempt to satisfy Condition 1. If we are able to

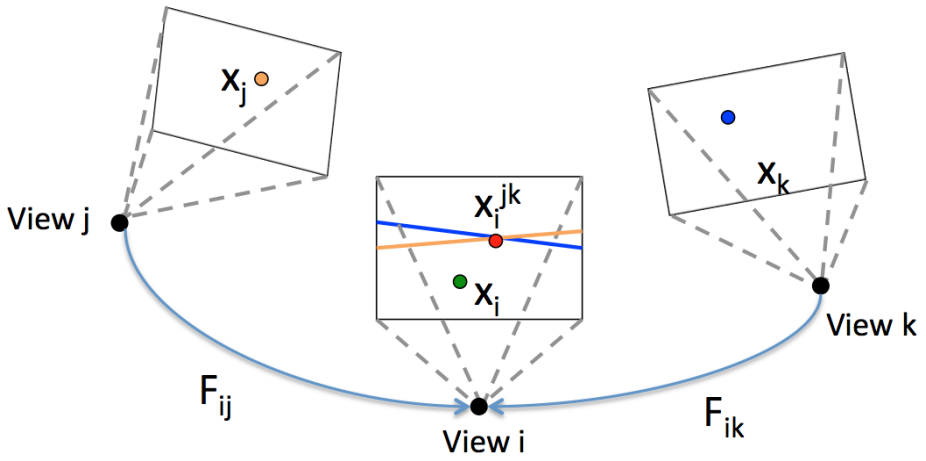

Figure 6.4: The epipolar point transfer is the intersection of the points x_j and x_k transferred to image i. We enforce loop consistency in the viewing graph by optimizing fundamental matrices such that the distance between the observed point x_i and the epipolar point transfer x_i^{jk} is minimized.

recover a consistent viewing graph then computing a reconstruction is trivial; however, even in the case that we cannot recover a fully consistent viewing graph the accuracy of the relative geometries improves enough that computing structure and motion is greatly simplified (*cf*. Section 6.5).

In the remainder of this section we propose an optimization that operates on the viewing graph, enforcing loop consistency with the epipolar point transfer. Our optimization recovers an approximately consistent viewing graph $\mathcal{G}_{OPT}$ that improves the performance of SfM by improving convergence in the estimation process.

6.4.1 Enforcing Loop Consistency

We now propose a cost function for adjusting $\mathcal{E}$ to enforce triplet consistency in $\mathcal{G}$. While Condition 1 is a sufficient condition for consistency [68], it is an algebraic metric and is significantly under-constrained. Instead, we propose to use the epipolar point transfer to enforce loop consistency. The epipolar point transfer is defined as the

intersection of two transfer lines of two views into a third view (*cf*. Figure 6.4).

$$\hat{x_i}^{jk} = F_{ij}x_j \times F_{ik}x_k \ , \tag{6.6}$$

where x_i is the feature point in image i and $\hat{x_i}^{jk}$ is the estimated pixel location of x_i based on the epipolar transfers from views j and k. In the ideal case we will have $x_i = \hat{x_i}^{jk}$; however, this is almost never the case in real data because of image noise and outliers in the feature matching process. Instead, we define a cost function based on the epipolar point transfer:

$$C(x)_i^{jk} = ||x_i - \hat{x_i}^{jk}||_2 \ . \tag{6.7}$$

This cost is a geometric error in terms of pixel distance and has previously been shown to be effective [28, 62]; however, care must be taken to avoid numerical instabilities (see Section 6.4.4).

6.4.2 Updating Fundamental Matrices

We seek to adjust fundamental matrix edges $F_{ij} \in \mathcal{E}$ in $\mathcal{G}$ based on Eq. (6.7). Fundamental matrices are a special class of rank-2 matrices [3]. Thus, updating a fundamental matrix during the nonlinear optimization must be done carefully to ensure that the resulting 3×3 matrix remains a valid fundamental matrix. We use the nonlinear fundamental matrix representation of Bartoli and Sturm [8] to update the fundamental matrices and briefly summarize their method here.

Note that a fundamental matrix F may be decomposed into matrices U, S, and V by singular value decomposition $F = USV^\top$, where U and V are orthonormal matrices and S is a 3×3 diagonal matrix of the form $diag(1, s, 0)$. To update F, we apply a $SO(3)$

rotation to the $O(3)$ matrices U and V, and a simple scalar addition to s.

$$U \leftarrow R_u U \tag{6.8}$$

$$V \leftarrow R_v V \tag{6.9}$$

$$s \leftarrow s + \delta_s \tag{6.10}$$

Since R_u and R_v are $SO(3)$ rotations, they may be represented with the minimal 3 parameters (by Euler angle or angle axis representation), thus requiring 7 parameters total (3 for R_u, 3 for R_v and 1 for δ_s) to update F. Since F has 7 degrees of freedom, this is a minimal parameterization and has been shown to maintain valid fundamental matrices [8].

6.4.3 Nonlinear Optimization

We create a large nonlinear optimization using the cost function of Eq. (6.7) and the presented method for updating fundamental matrices. We only optimize edges that are present in triplets $\mathcal{T}$ in the viewing graph:

$$\mathbf{F}^* = \arg\min_{F} \sum_{t \in \mathcal{T}} \sum_{x \in t} C(x)_i^{jk} + C(x)_j^{ik} + C(x)_k^{ij} \quad , \tag{6.11}$$

where x is a feature track present in the triplet $t = \{i, j, k\}$ and $\mathbf{F}$ is the set of fundamental matrices $F \in \mathcal{E}$. That is, for all triplets, we minimize the epipolar point transfer cost of all feature tracks within the triplet. Although the epipolar point transfer cost function does not require a triplet of fundamental matrices, we found that using triplets greatly improved the rate of convergence. Further, since each camera interacts with other cameras that might not be linked together in a triplet, larger loops are implicitly created.

Finally, it should be noted that the feature points x are treated as constant in

Eq. (6.11) and alternatively could be treated as free parameters that are optimized with the fundamental matrices. We found that additionally optimizing feature points with fundamental matrices resulted in a dramatic decrease in efficiency and did not provide significantly better results.

6.4.4 Numeric Instabilities

The epipolar point transfer has known degeneracies and numeric instabilities [33]. In particular, any configuration in which the transfer point lies on the trifocal plane of the images i, j, and k will be degenerate and points near this degeneracy are increasingly ill-conditioned. To avoid ill-conditioned points, we do not consider points where the two transfer lines are nearly parallel or when the transfer lines lay near the epipole. The latter scenario can be checked by examining the norm of the transfer line. Since the epipole is in the null space of F_{ij}, the norm of the transfer line will be very small when it is near the epipole.

It should be noted that if the three camera centers are collinear then there is a one-parameter family of planes containing the three cameras and thus the trifocal plane is ambiguous. We explicitly avoid this scenario by removing collinear triplets where the epipoles are equal. In practice, we did not find this to be a limitation since nearly all cameras in real datasets are constrained by at least one non-collinear camera triplet.

6.5 Estimating Structure and Motion

Given a consistent viewing graph, estimating structure and motion is extremely simple. To see why this is the case, let us consider a consistent and calibrated viewing graph $\mathcal{G}_C$. Since the graph is consistent, this means that the relative rotations in each triplet in $\mathcal{G}_C$ are also consistent (*i.e.*, concatenating the relative rotations in a triplet will form

Algorithm 2 Standard Global SfM Pipeline

Given: $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, Focal lengths

1: Filter $\mathcal{G}$ from loop constraints [58, 91]
2: Robust orientation estimation [14]
3: Filter edges by orientations [38]
4: Filter edges by relative translations [58, 88]
5: Estimate camera positions linearly [38] or nonlinearly [58, 88]
6: Triangulate 3D points
7: Bundle Adjustment

Return: 3D points X, and camera poses P

a loop: $R_{ij}R_{jk}R_{ki} = I$). The global orientations of each camera may be easily obtained from a random spanning tree [14] or from a linear orientation method [57]. A consistent viewing graph also means that the relative translation directions in $\mathcal{G}_C$ are perfect *i.e.*, $\alpha_{ij}t_{ij} = R_i(c_j - c_i)$. Thus, estimating the camera positions (assuming orientation is known) is equivalent to recovering the baselines α_{ij} between cameras. This pipeline is simpler than alternative global SfM approaches that require many filtering steps and more complex motion estimation algorithms [38, 58, 88] (*cf*. Algorithm 2).

While our viewing graph optimization is not guaranteed to create a consistent viewing graph, the optimization enforces enough of a consistency constraint that the SfM process can be simplified. In fact, we are able to remove all filtering steps from our SfM pipeline, and are able to further simplify the orientation and position estimation algorithms.

6.5.1 Viewing Graph Optimization

The viewing graph optimization described in Section 6.4 has $O(|\mathcal{E}|)$ free parameters, and thus the run time of the nonlinear optimization scales directly with the number of edges. Viewing graphs may contain highly redundant information, and so we would like to reduce the number of edges in the viewing graph so as to reduce the size of the nonlinear optimization. This is similar to the skeletal set selection of Snavely *et al.*

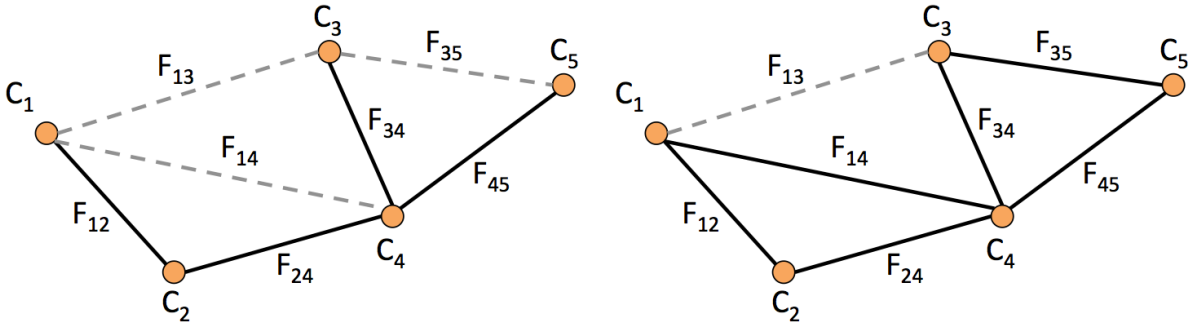

Figure 6.5: In order to reduce size of the viewing graph optimization, we construct a subgraph from the maximum spanning tree (MST). Edges in the MST (left) are shown with thick lines. Edges from the original viewing graph (dashed lines) are then added to the MST if they form a triplet to form $\mathcal{G}'$ (right).

[74], whose goal is to find a minimal set of views in the viewing graph that represent the entire scene. Our goal, in contrast, is to find a minimal set of edges that provide sufficient coverage over all views in the viewing graph.

Given an input viewing graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, we aim to create a subgraph $\mathcal{G}'$ that sufficiently covers the viewing graph with a minimum number of edges. Similar to [24], we first select the maximum spanning tree $\mathcal{G}' = \mathcal{G}_{MST}$ where edge weights are the number of inliers from fundamental matrix estimation between two views then find all edges $\mathcal{E}_T \in \mathcal{E}$ that, if added to $\mathcal{G}'$ would create a triplet in the graph (*i.e.*, a loop of size 3) as show in Figure 6.5. Among the edges in $\mathcal{E}_T$ we select a set of "good" edges $\mathcal{E}_G$ that have a triplet projection error less than τ (see Appendix B) and add these to the graph. The triplet projection error is an approximate error measurement to determine how close a triplet of fundamental matrices is to being consistent (Condition 1). We repeat this procedure (*i.e.*, $\mathcal{G}' = \mathcal{G}' \cup \mathcal{E}_G$) until every view in the viewing graph participates in at least one triplet, or there are no more "good" edges that can be added.

After we have obtained a representative viewing graph $\mathcal{G}'$, we must choose which feature tracks to use for the optimization. Similar to Crandall *et al.* [21], we use a set cover approach to select a subset of all feature tracks to accelerate optimization. In

Algorithm 3 Simplified Global SfM Pipeline

Given: $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$

1: Choose subgraph $\mathcal{G}'$ (Section 6.5.1)
2: Optimize the $\mathcal{G}'$ for consistency (Section 6.4)
3: [optional] Calibrate cameras (Section 6.2)
4: Estimate camera orientation from Eq. (6.12)
5: Estimate camera positions from Eq. (6.13)
6: Triangulate 3D points
7: Bundle Adjustment

Return: 3D points X, and camera poses P

each image, we create an $N \times N$ grid and choose the minimum number of feature tracks such that all grid cells in all images contain at least one track in the optimization. We have found that choosing spatially distributed feature points helps the viewing graph optimization to converge to a better minimum.

Finally, we use all selected edges and feature tracks to optimize the viewing graph by minimizing Eq. (6.11) using the Ceres Solver optimization library [4]. We use a Huber loss function to remain robust to outliers from feature matching.

6.5.2 Estimating Motion

The resulting optimized viewing graph provides accurate fundamental matrices that nearly form a consistent viewing graph (*cf*. Figure 6.6). As a result, there is no need for further outlier filtering during the structure and motion estimation. Further, there is no longer a need for robust methods such as [14] or [88]. This simplifies the SfM pipeline from a mathematical standpoint and for implementation purposes. The result is a more efficient pipeline with comparable accuracy to current methods.

Assuming the cameras are calibrated (or calibration is obtained with the method of Section 6.2), computing the orientations is simple. We solve for orientations by enforcing the relative rotation constraint $R_{ij} = R_j R_i^\top$. Similar to the method of [57], we minimize

the cost function

$$\sum_{i,j} ||R_i R_{ij} - R_j||_2 \tag{6.12}$$

to solve for camera orientations. Martinec and Pajdla [57] use a linear least squares technique to solve for matrices that minimize Eq. 6.12; however, this requires the solutions of the linear system to be projected into SO(3) matrices in order to obtain valid rotations. In contrast, we use the angle-axis parameterization (which ensures that all rotations R_i remain on the rotation manifold throughout the optimization[14]) and minimize Eq. (6.12) with a nonlinear solver. The orientations are initialized by chaining relative rotations from a random spanning tree as is done in the initialization for [14]. This simplified orientations solver is $2-4\times$ more efficient than the method of [14] while producing orientations that typically differ less than 1° for the datasets in Table 6.3.

To compute camera positions, we use the same nonlinear position constraint as Wilson and Snavely [88], though our pipeline does not require filtering steps before solving for camera positions. Given a relative translation t_{ij} and a known camera orientation R_i, we use the following constraint to estimate camera centers c_i and c_j:

$$t_{ij} = R_i \frac{(c_j - c_i)}{||c_j - c_i||} \ . \tag{6.13}$$

This nonlinear constraint is known to be more stable than other cross-product constraints [6, 29]. We use the Ceres Solver library [4] to solve the nonlinear Eq. (6.12) and Eq. (6.13) for recovering camera orientations and positions. After estimating camera poses, we triangulate 3D points and run a single bundle adjustment. Our SfM pipeline is summarized in Algorithm 3.

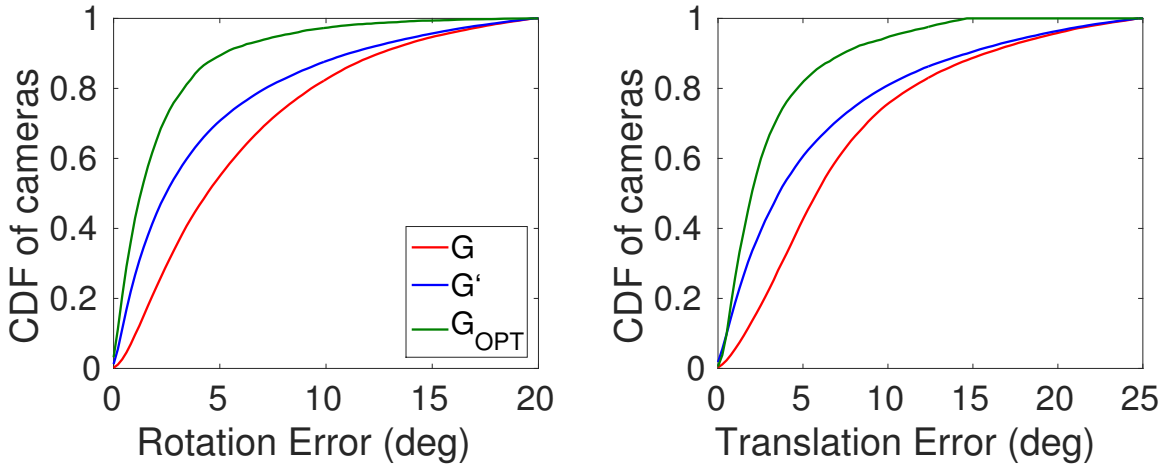

Figure 6.6: We plot the relative rotation and translation errors of the initial viewing graph $\mathcal{G}$, the subgraph $\mathcal{G}'$ and the viewing graph after optimization $\mathcal{G}_{OPT}$ when executed on the uncalibrated images from the Colosseum dataset [89]. The subgraph $\mathcal{G}'$ has lower relative pose errors than the initial viewing graph and the viewing graph optimization greatly improves the quality of relative poses.

6.6 Experimental Evaluation

We evaluate our algorithm on a number of small to large-scale benchmark datasets consisting of internet photo collections of popular landmarks. All experiments were performed on a 2008 Mac Pro with 2.26 GHz processor and 24 GB of RAM using a single core.

6.6.1 Viewing Graph Optimization

We demonstrate the effectiveness of our viewing graph optimization by examining the relative rotation and translation errors of the viewing graph compared to a reference reconstruction computed by VisualSFM[1]. The relative translation error is the angular distance (in degrees) between R_{ij} from the viewing graph and $R_j R_i^\top$ composed from the

[1]The reconstructions obtained with VisualSFM [89] are not meant to serve as ground truth but merely a reference for a good reconstruction.

reference reconstruction. Similarly, the relative translation error is the angular distance (in degrees) between the unit-norm vectors t_{ij} from the viewing graph and $\overline{t_{ij}} = (c_j - c_i)/||c_j - c_i||$ created from camera position c_j and c_i from the reference reconstruction. That is, $t_{err} = acos(t_{ij}^{\top}\overline{t_{ij}})$ is the translation error in degrees. We compare the relative pose errors on three different viewing graphs: the initial input viewing graph $\mathcal{G}$, the unoptimized subgraph $\mathcal{G}'$ (see Section 6.5.1), and the viewing graph after the viewing graph optimization $\mathcal{G}_{OPT}$.

The relative pose errors from the Colosseum dataset[89] are shown in Figure 6.6. The subgraph $\mathcal{G}'$ is effective in removing some of the inaccurate edges in $\mathcal{G}$; however, it is clear to see that our viewing graph optimization significantly improves the accuracy of relative poses. The mean relative rotation error on the Colosseum dataset is reduced from 8.3° in $\mathcal{G}$ to 7.5° in $\mathcal{G}'$ to 2.49° in $\mathcal{G}_{OPT}$. The mean relative translation error is reduced from 22.6° in $\mathcal{G}$ to 19.3° in $\mathcal{G}'$ to 3.29° in $\mathcal{G}_{OPT}$.

6.6.2 Focal Length Calibration

To determine the accuracy of our calibration method, we used images from the Pisa and Trevi datasets [38] that contain EXIF focal lengths and compare our calibration to reference focal lengths that were obtained from a reference reconstruction generated with VisualSfM [89] after bundle adjustment of the internal and external camera parameters. We compare our method to using EXIF data for calibration as well as the median focal length. The median focal length is obtained by decomposing all fundamental matrices connected to a view and taking the median of the focal lengths obtained from the decompositions.

We plot the accuracy of the focal lengths obtained with each method in Figure 6.7. For simplicity, we only plot the results from the Pisa dataset; however, the results from

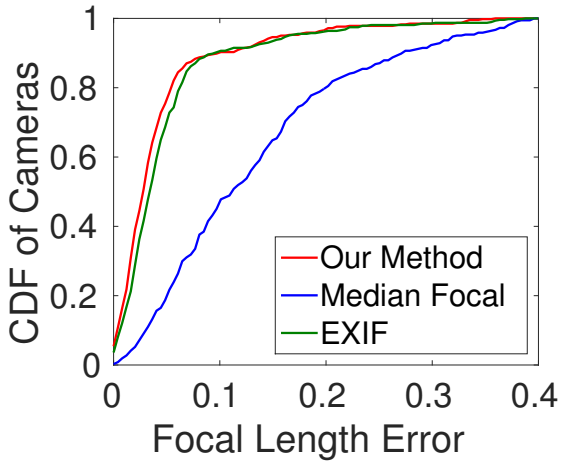

Figure 6.7: We show the accuracy of calibration methods on the Pisa dataset [38] and show the focal length error $|f - f_{gt}|/f_{gt}$ compared to ground truth focal lengths obtained from a reconstruction from VisualSfM [89]. Our method is at least as accurate as using EXIF, and is significantly more accurate than using the median focal lengths obtained from fundamental matrix decomposition.

the Trevi dataset were similar. For both datasets our calibration method converged in less than 10 seconds. Our method is at least as accurate as using focal length values from EXIF data. The accuracy stems from the use of many two-view constraints to estimate the focal length. EXIF values can be accurate but have the potential to be inaccurate if the image has been resized or cropped. Using the median focal length is very inaccurate and is not sufficient for use in a SfM pipeline.

6.6.3 Structure-from-Motion

We ran our pipeline on the small-scale dataset of [77] and the large-scale datasets of [88] to measure the performance and feasibility of our method on real data. We compare our SfM pipeline to several alternative global SfM pipelines, and the results are summarized in Tables 6.1, 6.2, 6.3, and 6.4.

Table 6.3 shows that our method is approximately up 2 to 10 times more efficient than alternative methods, while maintaining comparble accuracy to the state-of-the-art.

Table 6.1: We evaluate several SfM pipelines on the Strecha MVS datasets [77]. Our method shows excellent accuracy on all datasets.

	Accuracy (mm)				
Name	VSFM [89]	Olsson [60]	Cui *et al.* [22]	Moulon [58]	Ours
FountainP11	7.6	**2.2**	2.5	2.5	2.4
EntryP10	63.0	6.9	-	5.9	**5.7**
HerzJesuP8	19.3	3.9	-	**3.5**	**3.5**
HerzJesuP25	22.4	5.7	**5.0**	5.3	5.3
CastleP19	258	76.2	-	**25.6**	38.2
CastleP30	522	66.8	**21.2**	21.9	32.4

Table 6.2: Timing evaluation for several SfM methods on the Strecha MVS datasets [77]. Our method is the most efficient on nearly all datasets despite not using multiple threads as in [58] or a GPU as in [89]. Timing results of Cui *et al.* [22] were not available.

	Time (s)			
Name	VSFM [89]	Olsson [60]	Moulon [58]	Ours
FountainP11	**3**	133	5	4.5
EntryP10	**3**	88	5	3.8
HerzJesuP8	2	34	2	**1.9**
HerzJesuP25	12	221	10	**9.3**
CastleP19	9	99	6	**5.7**
CastleP30	18	317	14	**11.6**

The increase in efficiency is a direct result of our simplified SfM pipeline (see Section 6.5) that is able to efficiently utilize the high quality relative poses obtained from the optimized viewing graph. The statistical pose averaging (*cf*. Eq. (6.12) and Eq. (6.13)) converges to a high quality result very quickly because our optimized viewing graph is extremely accurate (*cf*. Figure 6.6).

Table 6.3: We compare results of several global SfM pipelines on the large-scale 1DSfM dataset [88]. We show the number of cameras reconstructed N_C and the median position error approximately in meters $\widetilde{x}$. For our method, $\widetilde{x}$ indicates position errors before bundle adjustment, and $\widetilde{x}_{BA}$ are the errors after bundle adjustment. Our method produces accurate camera poses before bundle adjustment and has comparable accuracy to alternative methods after bundle adjustment.

		1DSfM [88]		LUD [61]		Cui *et al.* [22]		Our Pipeline		
Name	N_c	N_c	$\widetilde{x}$	N_c	$\widetilde{x}$	N_c	$\widetilde{x}$	N_c	$\widetilde{x}$	$\widetilde{x}_{BA}$
Piccadilly	2152	1956	0.7	-	-	-	-	1928	5.2	1.0
Union Square	789	710	3.4	-	-	-	-	701	4.5	2.1
Roman Forum	1084	989	0.2	-	-	-	-	966	6.8	0.7
Vienna Cathedral	836	770	0.4	750	5.4	578	3.5	771	6.7	0.6
Piazza del Popolo	328	308	2.2	305	1.5	298	2.6	302	2.9	1.8
NYC Library	332	295	0.4	320	2.0	288	1.4	294	2.8	0.4
Alamo	577	529	0.3	547	0.4	500	0.6	533	1.4	0.4
Metropolis	341	291	0.5	-	-	-	-	272	8.7	0.4
Yorkminster	437	401	0.1	404	2.7	333	3.7	409	3.9	0.3
Montreal N.D.	450	427	0.4	433	0.5	426	0.8	416	2.0	0.3
Tower of London	572	414	1.0	425	4.7	393	4.4	409	9.3	0.9
Ellis Island	227	214	0.3	-	-	211	3.1	203	3.7	0.5
Notre Dame	553	507	1.9	536	0.3	539	0.3	501	9.4	1.2

6.7 Discussion

In this chapter, we have presented a new approach to large-scale SfM. Rather than focusing on creating potentially complex algorithms to overcome noise and outliers in the reconstruction process, we propose an optimization that corrects the viewing graph and enforces global consistency via loop constraints before applying SfM. We demonstrated that this optimization improves the quality of relative geometries in the viewing graph and removes the need for complex filtering steps as part of the SfM pipeline. Our viewing graph optimization works on calibrated or uncalibrated image sets and we provide a new method for calibrating cameras from a set of fundamental matrices. We incorporated the viewing graph optimization and focal length calibration into a global SfM pipeline that is intuitive to understand and easy to implement, and showed that this pipeline achieves greater efficiency and comparable accuracy to the current state-of-the-art methods. For

Table 6.4: Running time in seconds for the 1DSfM [88] experiment. T_{BA} and T_{Σ} denote the final bundle adjustment time and the total running times for each reconstruction method. T_{OPT} is the time our method takes for the viewing graph optimization. Our method is 2 to 9 times faster than alternative global SfM methods.

	1DSfM [88]		LUD [61]		Cui *et al.* [22]		Our Pipeline		
Name	T_{BA}	T_{Σ}	T_{BA}	T_{Σ}	T_{BA}	T_{Σ}	T_{OPT}	T_{BA}	T_{Σ}
Piccadilly	2425	3483	-	-	-	-	310	702	1246
Union Square	340	452	-	-	-	-	98	102	243
Roman Forum	1245	1457	-	-	-	-	284	847	1232
Vienna Cathedral	2837	3139	208	1467	717	959	139	422	607
Piazza del Popolo	191	249	31	162	93	144	12	78	101
NYC Library	392	468	54	200	48	90	14	83	154
Alamo	752	910	133	750	362	621	18	129	198
Metropolis	201	244	-	-	-	-	27	94	161
Yorkminster	777	899	148	297	63	108	13	71	102
Montreal N.D.	1135	1249	167	553	226	351	61	133	266
Tower of London	606	648	86	228	121	221	92	246	391
Ellis Island	139	171	-	-	64	95	12	14	33
Notre Dame	1445	1599	126	1047	793	1159	59	161	247

future work we plan to examine the guarantees we can make (if any) on the "consistency" of the viewing graph we obtain from the viewing graph optimization. Additionally, it would be interesting to see if our method may be applied for global SfM on projective reconstructions.

Figure 6.8: Visualizations of the Alamo (top-left), Yorkminster (top-right), and Tower of London (bottom) datasets from the 1DSfM experiments using internet photo collections.

Part III

Open Source Contributions and Conclusions

Chapter 7

Theia: A Fast and Scalable Structure-from-Motion Library

Structure-from-Motion (SfM) is a powerful tool capable of synthesizing a large amount of image information information into interesting 3D representations of scenes. In order to make research contributions to SfM, it is likely that inner steps of the pipeline must be altered or completely rethought; however, designing an SfM pipeline is a huge undertaking involving many non-trivial steps, complex algorithms, and many parameters to tune. While the benefits of SfM are clear, few publicly available tools exist for creating 3D reconstructions from SfM, and none of the currently available software is specifically geared towards *large-scale* SfM. Solutions such as the Bundler software that was popularized by the Photo Tourism work [73] existed but all available libraries only implemented incremental SfM pipelines, limiting the scalability. No software libraries implementing global SfM existed when I began my PhD. Further, I found the code for all of these libraries to be difficult to alter and extend with my own ideas due to poor code quality or outdated methods. Simply put, the solutions were great at creating high quality 3D models, but the available software was difficult for researchers to tweak and extend.

To address this issue, this chapter introduces the Theia library for Structure-from-Motion with the specific goal of making the library accurate, scalable, and most im-

portantly extendable. By taking a modular approach to the software design, the Theia library provides students, researchers, and industry experts with a clean C++ library including a state-of-the-art Structure-from-Motion pipeline and a vast collection of multi-view geometry tools and algorithms that utilize image and video inputs to create high quality 3D reconstructions. The library is BSD licensed with strict Google C++ style guide adherence for code readability and comprehensive unit test coverage. All algorithms are intentionally designed to be scalable, multithreaded computation is utilized whenever possible, and SSE optimization is enabled with the Eigen[1] linear algebra library. CMake[2] is used to ensure cross-platform portability and a simple build and installation process and Theia is well documented within the code and on the library's website[3]. Due to the modular software design, all algorithms can be easily extended, modified, or used independently of the rest of the library. Feature extraction, image matching, RANSAC, pose estimation and SfM methods may all be chosen at runtime, enabling simple experimentation and fine-tuning for obtaining high quality SfM reconstructions. Additionally, the library features a large number of tools for visualizing, evaluating, and comparing reconstructions. Since being released in February 2015 Theia has gathered an active community of users spanning graduate students, industry members, and computer vision experts.

7.1 Overview of Features

In this section we provide an overview of the core algorithms of Theia. The core features are modular so although they contribute to Theia's SfM pipeline, they may still be used independently. As such, the interface to the core features is typically generic

[1] `http://eigen.tuxfamily.org`
[2] `http://www.cmake.org`
[3] `http://www.theia-sfm.org`

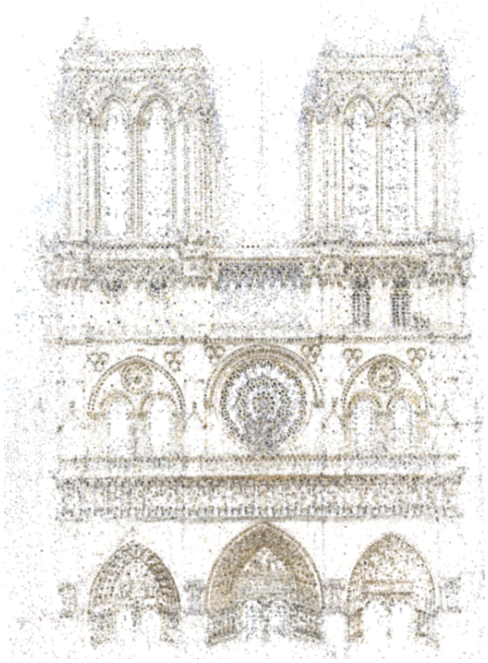

Figure 7.1: We are able to reconstruct Notre Dame from over 500 images in less than 120 seconds using Theia, compared to over 11 minutes using alternative software such as VisualSfM.

and not dependent on Theia-specific data types. The library contains a large number of useful algorithms for multi-view geometry, linear algebra, and optimization.

7.1.1 Comparison to other software

First, it is important to describe what makes Theia different from existing alternative libraries. There are several open-source SfM libraries available, but they differ from Theia in some key features. The Bundler library robustly computes 3D reconstructions with incremental SfM, limiting the ability to scale (see Section 6.5 for a comparison of incremental and global SfM). VisualSfM is a closed-source software that contains an incremental SfM pipeline that utilizes multi-threading and GPU programming for high efficiency; however, the scalability is still limited because of the incremental nature of the

pipeline. OpenMVG[4] is a SfM library with an active community that implements both an incremental SfM and global SfM pipeline; however, the focus of OpenMVG is on high accuracy for small to medium-sized problems and many of the methods were not designed with scalability in mind. Further, OpenMVG does not adhere to a consistent style guide and is not particularly modular, making it more difficult to extend and develop for casual users. The primary goals of the Theia library are usability, extendibility, and scalability.

7.1.2 Feature Extraction

Detecting salient image points is a fundamental aspect of computer vision. Feature detection and extraction methods that produce distinctive, repeatable image points and descriptors are desired for a wide range of applications such as object detection, image recognition, and multi-view stereo. In Theia, we implement a generic keypoint detector and feature descriptor extraction interface so that various types of image feature methods may be implemented and seamlessly integrated into the library. Further, this abstract interface allows the user to select the desired feature extraction method at run-time. The library currently contains implementations for SIFT features with support for AKAZE in development.

7.1.3 Feature Matching

To determine which images observe similar views of a scene, features are matched across images. There are currently two methods for feature matching in Theia: brute forces and a cascade hashing [15] that is over two orders of magnitude faster than brute force matching. Feature matching is also built with a generic interface so that new matching techniques may be seamlessly added and integrated into the library. The abstract

[4] `http://openmvg.readthedocs.org`

Table 7.1: Our RANSAC class is comprised of a `Sampler` class and a `QualityMeasurement` class that allow for alternative RANSAC approaches to be easily implemented. The table below outlines the RANSAC variants implemented in Theia by combining different sampling and quality measurement strategies.

	Random Sampler	**Progressive Sampler**	**EVT Sampler**	**SPRT Sampler**
Inlier	RANSAC [25]	PROSAC [17]	EVSAC [26]	-
MLE	MLESAC [83]	MLE + PROSAC	MLE + EVSAC	ARRSAC [65]

matching interface utilizes dynamic thread-pooling to optimize for multi-threaded performance, allowing Theia users to implement new matching techniques while getting the multi-threaded performance for free.

7.1.4 RANSAC

Random sample and consensus, or RANSAC, is one of the most commonly used algorithms in computer vision (*cf*. Chapter 2). As a result, much research has gone into making RANSAC extensions and variants that increase the efficiency or accuracy of the estimation. We have implemented a templated class that makes using RANSAC for estimation extremely easy as well as simple to extend. The user defines an `Estimator` class that estimates a model from a set of data. This allows the user to easily deploy any RANSAC class for a variety of tasks without having to rewrite the RANSAC-specific code.

Further, the RANSAC class itself is composed of an abstract `Sampler` class that samples the data and a `QualityMeasurement` class that determines how a model fits the data. For standard RANSAC, the `Sampler` class performs random sampling and the `QualityMeasurement` class counts the number of inliers in the data. These classes can be used to implement different RANSAC methods. For instance, using a maximum likelihood error as the `QualityMeasurement` would result in MLESAC[83] as shown in Table 7.1. RANSAC [25], PROSAC [17], MLESAC [83], ARRSAC [65], LMeds [67] and

EVSAC [26] have been implemented with this generic RANSAC interface.

7.1.5 Camera Pose Estimation

A fundamental problem in multi-view geometry is the ability to determine a camera's pose in a scene. This library implements numerous state-of-the-art pose estimation methods with generic interfaces so that they may be used independently of Theia's SfM pipeline. In addition to tcomprisedhe standard absolute and relative pose problems, we also implement methods for multi-camera systems, partially calibrated cameras, and scenarios where IMU information is available (*e.g.*, the vertical direction is known from IMU sensors). The following state-of-the-art solvers are currently implemented:

- Absolute pose: P3P [45], PnP [34]
- 4-point algorithm for absolute pose and focal length [12]
- 5-point algorithm for absolute pose, focal length, and radial distortion parameter(s) [46]
- Relative pose: 5-point essential matrix [75], 7-point fundamental matrix [33], 8-point fundamental matrix [32]
- Relative pose with known vertical direction: 3-point relative pose [78], 4-point for multi-camera systems [78]
- Homography from 4 points [33]
- Similarity transformations: from 3D-3D correspondences [86], from 2D-3D correspondences [79], from 2D-2D correspondences [80]

7.1.6 Global SfM Estimation Methods

The estimation step of global SfM pipelines can be broken into two parts: estimating camera orientations, and estimating camera positions. Both steps have received considerable research attention, though the position estimation problem is considered more difficult because positions do not form a Lie Group as rotations do. Combining different rotation and position methods can affect efficiency, accuracy, and robustness of global SfM pipelines. Theia takes a modular approach global SfM by using abstract `RotationEstimator` and `PositionEstimator` classes that define interfaces for estimating rotations and positions for global SfM, respectively. Rotation or position estimation methods may be easily constructed by deriving from these interfaces, allowing for seamless integration into the rest of the global SfM pipeline. This makes adding new estimation methods extremely simple and gives users the freedom to experiment with new methods without having to modify the entire pipeline. The rotation and position estimation choice may be set at run-time, allowing users to easily benchmark various estimation methods. Theia currently implements the following methods:

- Rotation Estimation: the robust method of Chatterjee *et al*. [14], the linear method of Martinec and Padjla [57], and a novel nonlinear method.
- Position Estimation: the least unsquared deviation formulation of Ozyesil and Singer [61], the nonlinear formulation of Wilson and Snavely [88], and the linear method of Jiang *et al*. [38].

7.1.7 Mathematics and Optimization

Multiple view geometry is centered around mathematics, and makes great use of fields such as linear algebra, probability, and optimization. While the mathematics are well-understood, implementing code to perform the mathematics can sometimes prove to be

more difficult. As such, Theia make a number of complex mathematics tools available as part of the library, in the hopes that they will be useful to other users.

- A scalable L_1 minimizer
- Polynomial solvers (closed form and iterative), including a novel SSE optimized re-implementation of RPOLY [36]
- Sparse matrix eigen-decomposition
- RQ matrix decomposition
- Bundle Adjustment
- Normalized graph cuts [70]

7.2 SfM Pipeline

At the core of Theia is the SfM module. Theia contains Incremental and Global SfM pipelines, but we only describe the Global SfM pipeline here as the incremental pipeline is similar to that proposed by Wu [89]. By combining the modular features presented in Section 7.1, we create SfM pipelines that are simple to follow, easily extendable, and highly scalable. The Incremental pipeline follows a standard sequential SfM procedure [73]. The Global SfM pipeline takes a set of pairwise relative poses between cameras as input, and outputs the orientation and position of all cameras in a global reference frame. The camera poses are computed through motion averaging algorithms. These global methods are inherently parallelizable and only require a single bundle adjustment, which is generally the most expensive part of SfM. This is in contrast to incremental SfM methods that add one new image at a time repeatedly perform bundle adjustment,

making them slower and less scalable than global SfM methods. Our global SfM pipeline is summarized with the following steps:

1. **Feature Extraction:** We extract feature descriptors (in parallel) at salient points within images. The feature type may be chosen at run-time for convenience.

2. **Image Matching:** After features are extracted, images must be matched to determine two-view geometry between images that observe the same scene. By default, Theia uses the extremely fast cascade hashing method [15] to compute image matches with multiple threads, though the matching technique may also be chosen at run-time.

3. **Estimate Camera Orientations:** We use the geometrically verified two-view matches from the previous step to estimate camera poses with global motion averaging schemes. A generic `RotationEstimator` abstract class is used to define the interface for rotation estimation methods. Currently, derived classes are implemented for three different rotation estimation methods: the robust orientation estimation algorithm [14], a linear L_2 averaging scheme [57], or a novel nonlinear estimation method.

4. **Estimate Camera Positions:** After camera orientations are estimated, the camera positions are estimated. Similar to the rotation estimation, we utilize an abstract `PositionEstimator` class to define the interface for position estimation methods. Currently, three methods are implemented for position estimation: a nonlinear position optimization [88], the linear method of [38], and the robust least unsquared deviations method of [61].

5. **Triangulate 3D Points:** After camera poses are estimated, 3D points are triangulated in parallel and refined with a nonlinear optimization.

6. **Bundle Adjustment:** As a final step, camera poses and 3D points are refined with a nonlinear optimization to minimize reprojection error. We use the Ceres Solver for scalable multi-threaded optimization to ensure high quality results are obtained efficiently.

7.2.1 ReconstructionBuilder

The simplest way to create a 3D reconstruction with Theia is to utilize the `ReconstructionBuilder` class. This class takes images as input and outputs 3D reconstructions created from the input images. The caller may choose to use Incremental or Global SfM at runtime. The options set in the `ReconstructionBuilder` control parameters for feature extraction, matching, pose estimation, triangulation, and bundle adjustment. Creating a reconstruction can be done in just a few lines of code:

```
ReconstructionBuilder builder(options);
for (const std::string& image : image_files)
    builder.AddImage(image);

std::vector<Reconstruction*> reconstructions;
builder.BuildReconstruction(&reconstructions);
```

7.2.2 Performance

Theia achieves state-of-the-art performance on large scale datasets both in terms of efficiency and accuracy. Timing results for several large scale datasets are shown in Table 7.2, and more results are available on the Theia website.

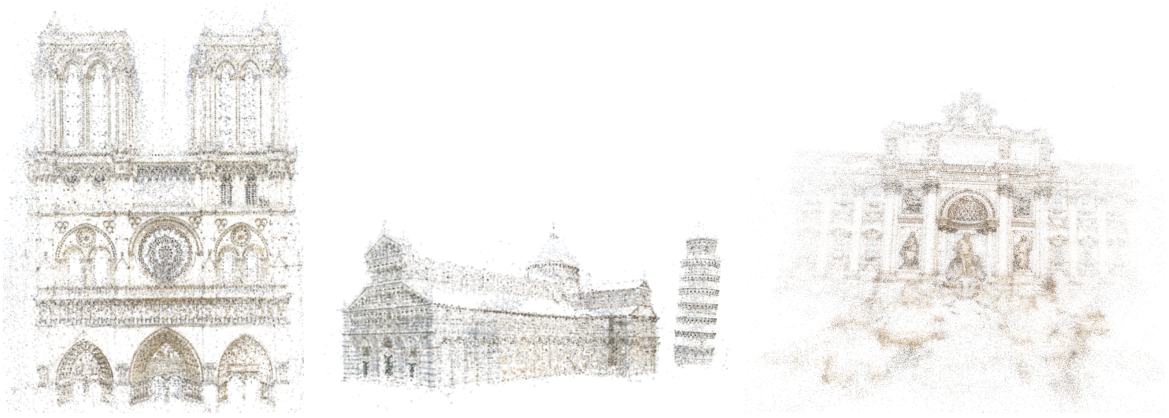

Figure 7.2: Theia's SfM pipeline is able to reconstruct 3D models for Notre Dame, Pisa, and the Trevi Fountain up to an order of magnitude faster than alternative open-source SfM libraries.

Table 7.2: Efficiency evaluation (in seconds) of Theia vs VisualSfM for large reconstructions using 8 threads. The number of images is given in parentheses for each dataset.

	VisualSfM	Theia
Notre Dame (553)	687	118
Pisa (481)	621	142
Trevi (1259)	2467	387

7.3 Impact

In this chapter, we have presented a comprehensive multi-view geometry library, Theia, that focuses on large-scale SfM. In addition to state-of-the-art scalable SfM pipelines, the library provides numerous tools that are useful for students, researchers, and industry experts in the field of multi-view geometry. Theia contains clean code that is well documented (with code comments and the website) and easy to extend. The modular design allows for users to easily implement and experiment with new algorithms within our current pipeline without having to implement a full end-to-end SfM pipeline themselves. Theia has already gathered a large number of diverse users from universities, startups, and industry and we hope to continue to gather users and active contributors from the open-source community.

Chapter 8

Conclusion

This thesis has presented my work creating detailed, explorable 3D models from with new scalable, accurate Structure-from-Motion (SfM) algorithms in spite of inaccurate or missing camera intrinsic calibration information. By explicitly developing the geometry and mathematics for the distributed camera these methods overcome limitations with previous SfM techniques so that image data can be crowd-source through internet photo collections to create 3D models of large, complex scenes that can be readily explored. This work makes the following specific contributions to the areas of computer vision and multiple view geometry:

- **Modeling the distributed camera.**
  - **The relative pose problem.** In Chapter 3, I examined the standard relative pose problem, and provided a generalization of this problem to the distributed camera. The generalized problem is equivalent to determining a similarity transformation between two sets of viewing rays that intersect in the general space. I have shown that the single-camera relative pose and multi-camera relative pose problems are specializations of this general form, and presented a solution to the generalized problem capable of solving any of the sub-problems.
  - **The absolute pose problem.** In Chapter 4, I investigated the absolute pose

problem of determining the pose of a camera from 2D-3D correspondences. As with Chapter 3, I show how to generalize this problem to the distributed camera, which is equivalent to compute the 7 degrees-of-freedom similarity transformation that aligns a set of viewing rays in one coordinate system to a set of corresponding 3D points in a different coordinate system by minimizing reprojection error. The solution to this problem is applicable for solving the single-camera and multi-camera absolute pose problems. I demonstrate how this method can be incorporated as a building block for a novel hierarchical SfM pipeline that is able to reconstruct nearly 16,000 images of Rome in just 20 minutes.

- **Calibrating the distributed camera.**
  - **A method for calibrating a camera from a 3D model** is discussed in Chapter 5. I present a method for computing a camera's pose and focal length by using importance sampling to intelligently guess the camera's focal length at each iteration of a RANSAC loop. The importance sampling is modified as the algorithm proceeds so that only the most likely focal lengths are sampled in successive iterations. This method is shown to give state-of-the-art performance on image-based localization tasks despite the absence of calibration information.
  - **A method for calibrating a network of cameras** that is capable of accurately and efficiently determining camera intrinsic parameters. Chapter 6 described this calibration procedure, and an efficient pre-SfM optimization that improves the efficiency of global SfM pipelines by up to an order of magnitude.

- **An open-source software library for Structure-from-Motion.** In Chapter 7, I describe my open-source SfM library that implements incremental and global SfM pipelines. The library is designed to be clean and easily extendable so that researchers and developers may easily experiment with specific parts of the SfM pipelines without having to worry about the rest of the pipeline. As such, it is a useful framework for researchers and developers interested in implementing new SfM techniques within a reliable, efficient framework. Many parameters may be easily set at run-time and Theia achieves state-of-the-art performance on both small and large-scale datasets. The Theia library was the winner of the 2015 ACM Multimedia Open Source Software Competition.

8.1 Future Work

In this thesis, I have presented methods for efficient and accurate large-scale SfM that are suitable for use on corrupted and diverse data sources such as internet photo collections. Using these methods, 3D geometry of large and complex scenes may be recovered and explored in a compelling way. While SfM models are an excellent mechanism for viewing large scenes, they are only able to recover sparse geometry and thus are not suitable for immersion. True immersion is simply not possible with sparse 3D models as it requires the scene to be viewed with the level of detail and completeness comparable to viewing it in the real world. As such, I view the contributions of this thesis as a first step towards full immersion in a scene and describe several future projects to help achieve this goal.

Virtual Reality is an extremely exciting medium for immersion with great potential to transform the way that humans explore the world and interact with each other. High-fidelity displays such as Oculus Rift provide a convincing experience of "being there"

and are able to transport us to entirely different places. An avenue of VR that greatly excites me is the potential ability to transport yourself anywhere in the world. Inspired by the vision of Noah Snavely [72] and others, I envision a *virtual tourism* system where users may type in a query location such as the Notre Dame Cathedral and instantly be transported to that location through immersive displays. Viewers may explore Notre Dame freely in the virtual environment, taking in it's vastness and appreciating the intricate details of the archways and the beautiful colors of the windows. This experience would be extremely compelling and would enable a wide variety of applications including virtual tourism, virtual field trips for students, and would even enable architectural and archaeological research with immersive viewing of scenes as they appeared in the past. Similarly, it would be interesting to view how a scene has changed over time with an immersive VR display.

Currently, real-world scenes are only suitable for viewing in VR displays when captured with multi-camera rigs that are able to recover sufficient multi-view stereo constraints. These rigs are prohibitively expensive for every-day consumers and require a careful capture process at the scenes of interest. What if we could leverage existing data to recover the necessary geometry instead? I believe SfM can be used for *content creation* for virtual reality by effectively crowd-sourcing public image data from the internet. If SfM is used as a first step for creating VR content, additional information such as dense detail, surface reconstruction, and texturing of the scene can be added to the sparse 3D model in order to create high-fidelity

Utilizing Video for SfM. Currently, video streams are an under-utilized source of rich visual information. Due to the temporal nature and high frame rate of videos, there is much more information that may be exploited to aid in 3D geometry extraction as compared to images alone. One of the challenges with video, however, is determining correspondence between different videos accurately and efficiently. One of the major

breakthroughs for SfM was the SIFT descriptor [55], which allowed correspondence to be determined between unordered images such as those found in internet photo collections. Prior to SIFT, SfM required ordered collections (such as frames from a single video stream) in order to determine correspondence between images. Currently, there is no adequate "SIFT-like" tool to determine correspondence between unordered videos from sources such as YouTube or Vimeo. While SIFT or other descriptors could be extracted from the individual frames of the video, this quickly becomes inefficient and, further, it ignores very powerful temporal information that videos contain.

Temporal information from videos could be utilized with simultaneous localization and mapping (SLAM) techniques. These techniques operate on individual video streams, resulting in a 3D map of the environment and a collection of posed keyframes from the videos. Each SLAM sequence could then be considered a distributed camera, and these distributed cameras could be merged into a common reference frame using either the methods of Chapter 3 or 4. Due to the sheer volume of data in videos, this procedure could prove challenging when considering many videos at once. Thus, the criteria for selecting keyframes in SLAM should be optimized such that only the most useful frames are saved. This would require a method for *temporal sampling* of live video streams to effectively compress the data.

Appendix A

Computing Matrices U, S, and V for Depth, Scale, and Translation

The constant matrices U, S, and V are used to recover depth, scale, and translation from the solution for the rotation matrix. These matrices are constructed as a function of known measurements from the matrix $(A^\top A)^{-1}A^\top$. Using the expression for A from Eq. (4.7), we have:

$$A^\top A = \left[\begin{array}{ccc|cc} 1 & & & \bar{r_1}^\top q_1 & -\bar{r_1}^\top \\ & \ddots & & \vdots & \vdots \\ & & 1 & \bar{r_n}^\top q_n & -\bar{r_n}^\top \\ \hline q_1^\top \bar{r_1} & \dots & q_n^\top \bar{r_n} & \sum_{i=1}^n q_i^\top q_i & \sum_{i=1}^n -q_i^\top \\ -\bar{r_1} & \dots & -\bar{r_n} & \sum_{i=1}^n -q_i & nI \end{array}\right] = \left[\begin{array}{c|c} \mathcal{A} & \mathcal{B} \\ \hline \mathcal{B}^\top & \mathcal{D} \end{array}\right], \tag{A.1}$$

where solid lines represent the block-matrix boundaries. Through block matrix inversion, we can conveniently solve for the inverse:

$$
\begin{aligned}
(A^\top A)^{-1} &= \left[\begin{array}{c|c} \mathcal{E} & \mathcal{F} \\ \hline \mathcal{G} & \mathcal{H} \end{array}\right] \\
\mathcal{E} &= I + \mathcal{B}\mathcal{H}\mathcal{B}^\top \\
\mathcal{F} &= -\mathcal{B}\mathcal{H} \\
\mathcal{G} &= -\mathcal{H}\mathcal{B}^\top \\
\mathcal{H} &= \left(\begin{bmatrix} \sum_{i=1}^n q_i^\top q_i & \sum_{i=1}^n -q_i^\top \\ \sum_{i=1}^n -q_i & nI \end{bmatrix} - \begin{bmatrix} \sum_{i=1}^n q_i^\top \bar{r}_i \bar{r}_i{}^\top q_i & \sum_{i=1}^n -q_i^\top \bar{r}_i \bar{r}_i{}^\top \\ \sum_{i=1}^n -\bar{r}_i \bar{r}_i{}^\top q_i & \sum_{i=1}^n \bar{r}_i \bar{r}_i{}^\top \end{bmatrix} \right)^{-1}.
\end{aligned}
\tag{A.2}
$$

Finally, we can compute U, S, and V from Eq. (A.2). Many of the terms can be simplified because of multiplications involving $\bar{r}_i{}^\top \bar{r}_i = 1$. This leaves us with a greatly simplified

expression for U, S, and V:

$$\left[\begin{array}{c} U \\ \hline S \\ \hline V \end{array}\right] = (A^\top A)^{-1} A^\top$$

$$U = \begin{bmatrix} \bar{r_i}^\top & & \\ & \ddots & \\ & & \bar{r_n}^\top \end{bmatrix} + \mathcal{B} \begin{bmatrix} S \\ V \end{bmatrix} \tag{A.3}$$

$$\left[\begin{array}{c} S \\ \hline V \end{array}\right] = -\mathcal{H}\mathcal{B}^\top \begin{bmatrix} \bar{r_i}^\top & & \\ & \ddots & \\ & & \bar{r_n}^\top \end{bmatrix} + \mathcal{H} \begin{bmatrix} q_1^\top & \dots & q_n^\top \\ -I & \dots & -I \end{bmatrix}$$

$$= \mathcal{H} \begin{bmatrix} q_1^\top - q_1^\top \bar{r_1} \bar{r_1}^\top & \dots & q_n^\top - q_n^\top \bar{r_n} \bar{r_n}^\top \\ \bar{r_1}\bar{r_1}^\top - I & \dots & \bar{r_n}\bar{r_n}^\top - I \end{bmatrix}.$$

Appendix B

Triplet Projection Error

We define here the triplet projection error used in Section 6.5.1. Given three views, i, j, and k, and the corresponding fundamental matrices F_{ij}, F_{ik}, and F_{jk}, Sinha *et al.* [71] compute a consistent triplet of fundamental matrices. We use their technique to define a triplet projection error that measures the consistency of a triplet of fundamental matrices. We will briefly summarize the method here.

First, projection matrices for views i and j and k are constructed from the fundamental matrices

$$P_i = [I|0] \tag{B.1}$$

$$P_j = [[e_{ji}]_\times F_{ij}|e_{ji}] \tag{B.2}$$

$$P_k = [[e_{ki}]_\times F_{ik}|0] + e_{ki}v^\top \tag{B.3}$$

where v is an unknown 4-vector. Recall from [33] that a fundamental matrix may be constructed from the projection matrices of the two views it connects:

$$\overline{F}_{jk}^\top = [e_{kj}]_\times P_k P_j^+. \tag{B.4}$$

$\overline{F}_{jk}$ is linear in v and all possible solutions for $\overline{F}_{jk}$ span the subspace of possible funda-

mental matrices that will form a consistent triplet as defined in Condition (1) [71]. We solve for v that yields $\overline{F}_{jk}$ closest to F_{jk}. We define the triplet projection error as the difference of $\overline{F}_{jk}$ and F_{jk} by Frobenius norm:

$$Err_{ijk} = ||\overline{F}_{jk} - F_{jk}|| \ . \tag{B.5}$$

Bibliography

[1] M. A. Abidi and T. Chandra, *A New Efficient and Direct Dolution for Pose Estimation Using Quadrangular Targets: Algorithm and Evaluation*, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **17** (1995), no. 5 534–538.

[2] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski, *Building rome in a day*, *Communications of the ACM* **54** (2011), no. 10 105–112.

[3] S. Agarwal, H.-l. Lee, B. Sturmfels, and R. R. Thomas, *Certifying the existence of epipolar matrices*, *arXiv preprint arXiv:1407.5367* (2014).

[4] S. Agarwal, K. Mierle, and Others, "Ceres solver." `http://ceres-solver.org`.

[5] S. Agarwal, N. Snavely, S. M. Seitz, and R. Szeliski, *Bundle adjustment in the large*, in *European Conference on Computer Vision*, pp. 29–42. Springer, 2010.

[6] M. Arie-Nachimson, S. Z. Kovalsky, I. Kemelmacher-Shlizerman, A. Singer, and R. Basri, *Global motion estimation from point matches*, in *International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT)*, pp. 81–88, IEEE, 2012.

[7] E. Ask, K. Yubin, and K. Astrom, *Exploiting p-fold symmetries for faster polynomial equation solving*, in *Proceedings of the International Conference on Pattern Recognition (ICPR)*, IEEE, 2012.

[8] A. Bartoli and P. Sturm, *Nonlinear estimation of the fundamental matrix with minimal parameters*, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **26** (March, 2004) 426–432.

[9] B. Bhowmick, S. Patra, A. Chatterjee, V. Govindu, and S. Banerjee, *Divide and conquer: Efficient large-scale structure from motion using graph partitioning*, in *The Asian Conference on Computer Vision*, pp. 273–287. Springer International Publishing, 2015.

[10] M. Bujnak, Z. Kukelova, and T. Pajdla, *Robust Focal Length Estimation by Voting in Multi-view Scene Reconstruction*, in *Proceedings of the Asian Conference on Computer Vision (ACCV)*, 2009.

[11] M. Bujnak, Z. Kukelova, and T. Pajdla, *Making Minimal Solvers Fast*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[12] M. Bujnak, Z. Kukelova, and T. Pajdla, *A general solution to the p4p problem for camera with unknown focal length*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8, IEEE, 2008.

[13] M. Bujnak, Z. Kukelova, and T. Pajdla, *3d reconstruction from image collections with a single known focal length*, in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 1803–1810, IEEE, 2009.

[14] A. Chatterjee and V. M. Govindu, *Efficient and robust large-scale rotation averaging*, in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 521–528, IEEE, 2013.

[15] J. Cheng, C. Leng, J. Wu, H. Cui, and H. Lu, *Fast and accurate image matching with cascade hashing for 3d reconstruction*, in *CVPR*, IEEE, 2014.

[16] O. Chum and J. Matas, *Randomized ransac with td, d test*, in *Proceedings of the British Machine Vision Conference (BMVC)*, vol. 2, pp. 448–457, 2002.

[17] O. Chum and J. Matas, *Matching with PROSAC-progressive sample consensus*, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[18] O. Chum and J. Matas, *Optimal randomized ransac*, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **30** (2008), no. 8 1472–1482.

[19] O. Chum, J. Matas, and J. Kittler, *Locally optimized ransac*, in *Pattern Recognition*, pp. 236–243. Springer, 2003.

[20] D. Crandall, A. Owens, N. Snavely, and D. Huttenlocher, *SfM with MRFs: Discrete-continuous optimization for large-scale structure from motion*, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* **35** (December, 2013) 2841–2853.

[21] D. Crandall, A. Owens, N. Snavely, and D. P. Huttenlocher, *Discrete-continuous optimization for large-scale structure from motion*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[22] Z. Cui, N. Jiang, and P. Tan, *Linear global translation estimation from feature tracks*, *arXiv preprint arXiv:1503.01832* (2015).

[23] C. Engels, H. Stewénius, and D. Nistér, *Bundle adjustment rules*, *Photogrammetric computer vision* **2** (2006) 124–131.

[24] O. Enqvist, F. Kahl, and C. Olsson, *Non-sequential structure from motion*, in *Computer Vision Workshops of the IEEE International Conference on Computer Vision (ICCV)*, pp. 264–271, IEEE, 2011.

[25] M. A. Fischler and R. C. Bolles, *Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography*, *Communications of the ACM* **24** (1981), no. 6 381–395.

[26] V. Fragoso, P. Sen, S. Rodriguez, and M. Turk, *EVSAC: Accelerating Hypotheses Generation by Modeling Matching Scores with Extreme Value Theory*, in *Proceedings of IEEE International Conference on Computer Vision*, IEEE, 2013.

[27] J.-M. Frahm, P. Fite-Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik, and M. Pollefeys, *Building rome on a cloudless day*, in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2010.

[28] A. Goldstein and R. Fattal, *Video stabilization using epipolar geometry*, *ACM Transactions on Graphics (TOG)* **31** (2012), no. 5 126.

[29] V. M. Govindu, *Combining two-view constraints for motion estimation*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, pp. II–218, IEEE, 2001.

[30] V. M. Govindu, *Lie-algebraic averaging for globally consistent motion estimation*, in *Proceeding of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. I–684, IEEE, 2004.

[31] R. Haralick, C.-N. Lee, K. Ottenberg, and M. Nölle, *Review and analysis of solutions of the three point perspective pose estimation problem*, *International Journal of Computer Vision* **13** (1994), no. 3 331–356.

[32] R. Hartley, *In defense of the eight-point algorithm*, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **19** (1997), no. 6 580–593.

[33] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[34] J. Hesch and S. Roumeliotis, *A direct least-squares (dls) solution for pnp*, in *Proceedings of the International Conference on Computer Vision (ICCV)*, IEEE, 2011.

[35] A. Irschara, C. Zach, J.-M. Frahm, and H. Bischof, *From Structure-from-Motion Point Clouds to Fast Location Recognition*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[36] M. A. Jenkins, *Algorithm 493: Zeros of a real polynomial [c2]*, *ACM Transactions on Mathematical Software (TOMS)* **1** (1975), no. 2 178–189.

[37] Y. Jeong, D. Nister, D. Steedly, R. Szeliski, and I.-S. Kweon, *Pushing the envelope of modern methods for bundle adjustment*, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **34** (2012), no. 8 1605–1617.

[38] N. Jiang, Z. Cui, and P. Tan, *A global linear method for camera pose registration*, in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 481–488, IEEE, 2013.

[39] K. Kanatani, *Group-theoretical methods in image understanding*, vol. 2. springer-Verlag New York, 1990.

[40] K. Kanatani and C. Matsunaga, *Closed-form expression for focal lengths from the fundamental matrix*, in *Proceedings of the Asian Conference on Computer Vision*, vol. 1, pp. 128–133, 2000.

[41] Y. Kanazawa, Y. Sugaya, and K. Kanatani, *Decomposing three fundamental matrices for initializing 3-d reconstruction from three views*, *IPSJ Transactions on Computer Vision and Applications* **6** (2014) 120–131.

[42] G. Klein and D. Murray, *Parallel tracking and mapping for small ar workspaces*, in *IEEE International Symposium on Mixed and Augmented Reality*, pp. 225–234, IEEE, 2007.

[43] L. Kneip and H. Li, *Efficient computation of relative pose for multi-camera systems*, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, IEEE, 2014.

[44] L. Kneip, H. Li, and Y. Seo, *Upnp: An optimal o (n) solution to the absolute pose problem with universal applicability*, in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 127–142. Springer International Publishing, 2014.

[45] L. Kneip, D. Scaramuzza, and R. Siegwart, *A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2969–2976, IEEE, 2011.

[46] Z. Kukelova, M. Bujnak, and T. Pajdla, *Real-Time Solution to the Absolute Pose Problem with Unknown Radial Distortion and Focal Length*, in *Proceedings of the International Conference on Computer Vision*, 2013.

[47] Z. Kukelova, M. Bujnak, and T. Pajdla, *Automatic generator of minimal problem solvers*, in *European Conference on Computer Vision*, pp. 302–315. Springer, 2008.

[48] A. Kushal and S. Agarwal, *Visibility based preconditioning for bundle adjustment*, in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1442–1449, IEEE, 2012.

[49] V. Lepetit, F. Moreno-Noguer, and P. Fua, *Epnp: An accurate o (n) solution to the pnp problem*, *International Journal of Computer Vision* **81** (2009), no. 2 155–166.

[50] H. Li, R. Hartley, and J.-h. Kim, *A linear approach to motion estimation using generalized camera models*, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, IEEE, 2008.

[51] Y. Li, N. Snavely, D. Huttenlocher, and P. Fua, *Worldwide Pose Estimation Using 3D Point Clouds*, in *Proceedings of the European Conference on Computer Vision*, 2012.

[52] Y. Li, N. Snavely, and D. Huttenlocher, *Location recognition using prioritized feature matching*, in *Proceedings of the European Conference on Computer Vision*, vol. 6312, pp. 791–804. Springer, 2010.

[53] M. I. Lourakis, A. Argyros, *et. al.*, *Is levenberg-marquardt the most efficient optimization algorithm for implementing bundle adjustment?*, in *IEEE International Conference on Computer Vision*, vol. 2, pp. 1526–1531, IEEE, 2005.

[54] M. I. Lourakis and A. A. Argyros, *Sba: A software package for generic sparse bundle adjustment*, *ACM Transactions on Mathematical Software (TOMS)* **36** (2009), no. 1 2.

[55] D. G. Lowe, *Distinctive image features from scale-invariant keypoints*, *International Journal of Computer Vision* **60** (2004), no. 2 91–110.

[56] F. Macaulay, *Some formulae in elimination*, *London Mathematical Society* **1** (1902), no. 1 3–27.

[57] D. Martinec and T. Pajdla, *Robust rotation and translation estimation in multiview reconstruction*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8, June, 2007.

[58] P. Moulon, P. Monasse, and R. Marlet, *Global Fusion of Relative Motions for Robust, Accurate and Scalable Structure from Motion*, in *Proceedings of IEEE International Conference on Computer Vision*, 2013.

[59] D. Nistér, *An efficient solution to the five-point relative pose problem*, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **26** (2004), no. 6 756–770.

[60] C. Olsson and O. Enqvist, *Stable structure from motion for unordered image collections*, in *Image Analysis*, pp. 524–535. Springer, 2011.

[61] O. Ozyesil and A. Singer, *Robust camera location estimation by convex programming*, in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June, 2015.

[62] J. K. Pillai, *Consistent averaging of multi-camera epipolar geometries*, Master's thesis, India Institute of Science, 2008.

[63] R. Pless, *Using many cameras as one*, in *Proceedings IEEE Conference on Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. II–587, IEEE, 2003.

[64] J. Plùcker, *On a new geometry of space*, *Philosophical Transactions of the Royal Society of London* **155** (1865) 725–791.

[65] R. Raguram, J.-M. Frahm, and M. Pollefeys, *A comparative analysis of ransac techniques leading to adaptive real-time random sample consensus*, in *ECCV*. Springer, 2008.

[66] R. Raguram, J.-M. Frahm, and M. Pollefeys, *Exploiting Uncertainty in Random Sample Consensus*, in *Proceedings of the International Conference on Computer Vision*, 2009.

[67] P. J. Rousseeuw, *Least median of squares regression*, *Journal of the American statistical association* **79** (1984), no. 388 871–880.

[68] A. Rudi, M. Pizzoli, and F. Pirri, *Linear solvability in the viewing graph*, in *Proceedings of the Asian Conference on Computer Vision*, pp. 369–381. Springer, 2011.

[69] T. Sattler, B. Leibe, and L. Kobbelt, *Improving Image-Based Localization by Active Correspondence Search*, in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012.

[70] J. Shi and J. Malik, *Normalized cuts and image segmentation*, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* **22** (2000), no. 8 888–905.

[71] S. N. Sinha, M. Pollefeys, and L. McMillan, *Camera network calibration from dynamic silhouettes*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, pp. I–195, IEEE, 2004.

[72] K. N. Snavely *et. al.*, *Scene reconstruction and visualization from internet photo collections*, .

[73] N. Snavely, S. M. Seitz, and R. Szeliski, *Photo tourism: exploring photo collections in 3d*, in *ACM transactions on graphics (TOG)*, vol. 25, pp. 835–846, ACM, 2006.

[74] N. Snavely, S. M. Seitz, and R. Szeliski, *Skeletal graphs for efficient structure from motion.*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, p. 2, 2008.

[75] H. Stewenius, C. Engels, and D. Nistér, *Recent developments on direct relative orientation*, *ISPRS Journal of Photogrammetry and Remote Sensing* **60** (2006), no. 4 284–294.

[76] H. Stewénius, D. Nistér, M. Oskarsson, and K. Åström, *Solutions to minimal generalized relative pose problems*, in *Workshop on Omnidirectional Vision*, 2005.

[77] C. Strecha, W. von Hansen, L. V. Gool, P. Fua, and U. Thoennessen, *On benchmarking camera calibration and multi-view stereo for high resolution imagery*, in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1–8, IEEE, 2008.

[78] C. Sweeney, J. Flynn, and M. Turk, *Solving for relative pose with a partially known rotation is a quadratic eigenvalue problem*, in *Proceedings of the International Conference on 3D Vision*, IEEE, 2014.

[79] C. Sweeney, V. Fragoso, T. Hollerer, and M. Turk, *gdls: A scalable solution to the generalized pose and scale problem*, in *European Conference on Computer Vision*, vol. 8692, pp. 16–31, Springer, 2014.

[80] C. Sweeney, L. Kneip, T. Hollerer, and M. Turk, *Computing similarity transformations from only image correspondences*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3305–3313, 2015.

[81] R. Szeliski, *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[82] F. Tisseur and K. Meerbergen, *The quadratic eigenvalue problem*, *SIAM review* **43** (2001), no. 2 235–286.

[83] P. H. Torr and A. Zisserman, *MLESAC: A new robust estimator with application to estimating image geometry*, *Computer Vision and Image Understanding* **78** (2000), no. 1 138–156.

[84] B. Triggs, *Camera Pose and Calibration from 4 or 5 Known 3D Points*, in *Proceedings of the International Conference on Computer Vision*, 1999.

[85] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, *Bundle adjustmenta modern synthesis*, in *Vision algorithms: theory and practice*, pp. 298–372. Springer, 2000.

[86] S. Umeyama, *Least-squares estimation of transformation parameters between two point patterns*, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **13** (1991), no. 4 376–380.

[87] J. Ventura, C. Arth, G. Reitmayr, and D. Schmalstieg, *A minimal solution to the generalized pose-and-scale problem*, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2014.

[88] K. Wilson and N. Snavely, *Robust global translations with 1dsfm*, in *Proceedings of the European Conference on Computer Vision*, pp. 61–75. Springer, 2014.

[89] C. Wu, *Towards linear-time incremental structure from motion*, in *Proceedings of the International Conference on 3D Vision*, pp. 127–134, IEEE, 2013.

[90] C. Wu, S. Agarwal, B. Curless, and S. M. Seitz, *Multicore bundle adjustment*, in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3057–3064, IEEE, 2011.

[91] C. Zach, M. Klopschitz, and M. Pollefeys, *Disambiguating visual relations using loop constraints*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1426–1433, IEEE, 2010.

[92] Y. Zheng, Y. Kuang, S. Sugimoto, K. Astrom, and M. Okutomi, *Revisiting the pnp problem: A fast, general and optimal solution*, in *Proceedings of the International Conference on Computer Vision*, IEEE, Dec, 2013.